型番:MSC-MOD60　　REV.3.30
開発元型式:uALFAT-SD

SDカード簡単シリアルアクセスボード

取扱説明書
お使いになる前にこの説明書をよくお読みの上正しくお使いください。本製品のサポートは製品の開発元である米GHI Electronics社が行います。

(C)2009 マイクロテクニカ

はじめに本マニュアルについて

この度は、SDカードUARTアクセスモジュール(型式:MSC-MOD60、以下型式で記載)をご利用頂き、誠にありがとうございます。

本マニュアルは、本製品の開発元である米GHI Electronics社のマニュアルを元に、当方が日本語訳や解説等を追加して作成された日本語マニュアルです。英語版マニュアルではわかりにくい部分を図解で解説するなどしてマニュアルとして読みやすく編集しておりますが、すべての部分において完全な日本語訳となっていない部分がございます。

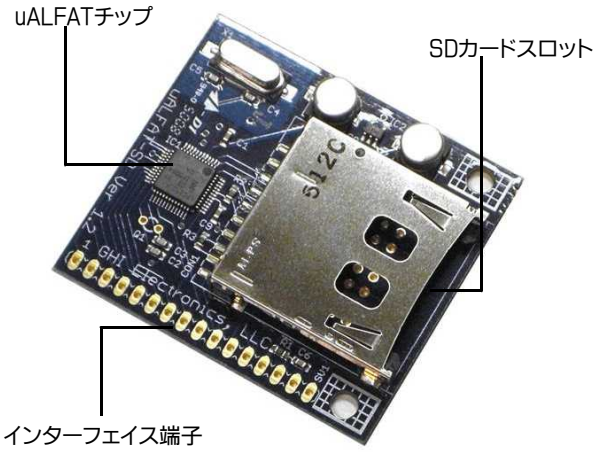
翻訳に誤りが存在する場合もございますので、開発元よりリリースされている最新の英語版マニュアルを必ずご参照頂き、本日本語マニュアルと併せてお読み頂けますようお願い申し上げます。

■開発元のオリジナルマニュアルダウンロード先
http://www.ghielectronics.com/downloads/uALFAT/uALFAT%20Manual.pdf

※当方のマニュアルダウンロードページ(http://www.microtechnica.net/)からもダウンロードできます。

なお、日本語マニュアル作成においては適正かつ正確なものとなるよう注意を払っておりますが、場合によっては正当性や妥当性に不確実な部分が含まれていることもございます。日本語マニュアルに記載されている情報の利用による損害が発生しても当方では責任を負いかねますので予めご了承下さい。

製品本体



製品の概要

SDカード簡単シリアルアクセスボード(型式:MSD-MOD60、以下型式で記載)は、MMC/SDカードスロットを本体に搭載し、MMCやSDカードに簡単なシリアル通信コマンドで、アクセスすることのできるボードです。FATファイルシステムを搭載しています。

MSC-MOD60はFAT16、FAT32に対応しており、ファイルの新規作成や更新、ディレクトリの作成、ファイルやディレクトリの削除などファイル操作を簡単に行うことができます。ファイルシステムはWindowsと同じFAT形式ですので、MSC-MOD60で作成、更新したファイルはそのままWindowsで扱うことができます。

MSC-MOD60ではロングファイルネーム(LFN)に完全対応しました。ファイル名やディレクトリ名の指定にロングファイルネームが使えます。

MSC-MOD60は高速処理を実現しています。シリアル通信側は最大で921600bpsまで通信速度を設定できます。もちろん高速通信時でも待ち時間はありません。ノンウェイトでこれだけの速度を出せますので大きなファイルの作成や読み込みも楽々行えます。

速度よりも消費電力を小さくしたいというニーズにもお応えできるようパワーモードを搭載しています。フルパワーモード、省電力モード、また最低の設定条件を保持したまま休止状態に移行するハイバネートモードを搭載しており、お客様のアプリケーションにあったシステムを構築することができます。

付属の32.768KHzクリスタル発振子を取り付けることで、作成したファイルのタイムスタンプをつけることができます。バッテリーバックアップ機能付きですので外部にコイン電池などを取り付けることで電源切断時も時計計測を継続できます。

パッケージの内容

■同梱物
・MSC-MOD60本体
・32.768KHzクリスタル発振子
・マニュアル(本書)　※マニュアルはダウンロードにて提供

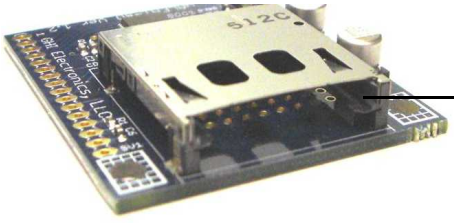
対応メモリーカードと対応ファイルシステム

MSC-MOD60には最小128MB～最大16GBまでの一般的に販売されているSDカードが装着できます。

対応するファイルシステムは、Windows標準のFAT16及びFAT32です。NTFSには対応しておりません。

■メモリーカードの装着/取り外し
SDカードは、MSC-MOD60基板にあるソケットに装着します。
装着の際には、SDカードの端子部分が基板側(下側)になるような方向で挿入し、"カチッ"と音がするまで押し込んでください。

なお、コネクターの仕様上、SDカードが2枚入るように見えますが、挿入できるのは1枚だけです。挿入する場合には必ず下図の通り、上側に挿入してください。



取り外す際には、カードを引っぱらずに、一度押し込んむとカードが飛び出しますので、それから引き抜いてください。

対応できるマイクロSDカードと相性問題について

MSC-MOD60には家電量販店等で市販されている、SDカードが使用できます。しかしながら現在SDカードは国内外の多くのメーカーから多種多様の製品が販売されており、希ではありますが条件等によっては相性問題が発生する場合があります。

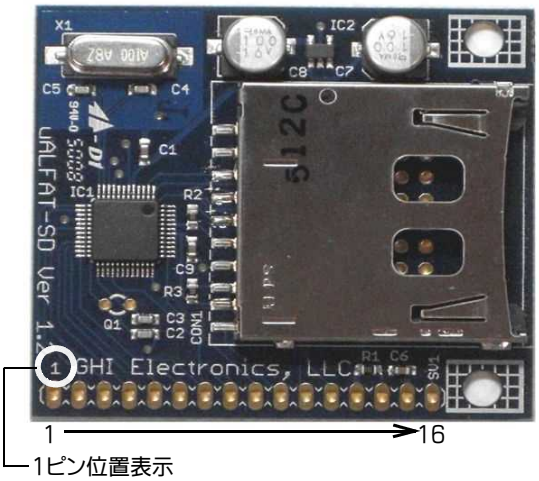
相性問題の多くは使用されているメモリーICの種類やコントローラーのファームウエア等によって時間的なタイミングがずれていたりすることによります。それぞれの製品では規格を満たしていても、規格にはある程度の余裕が設けられているため、その余裕が積み重なり時間的なタイミングのずれが重畳されて大きくなり、相性の問題として不具合を引き起こす原因となります。これは、多くのメーカーが参入している開かれた規格では技術的にも不可避的です。

上記のような理由により、誠に勝手ではありますが、相性問題によって動作に不具合が生じた場合であっても当方では、その責を負うことはできませんので、ご理解とご了承お願い申し上げます。

MSC-MOD60の扱うことのできるSDカードは最低容量128MB以上となっておりますので、合わせてご注意ください。

インターフェイス端子の概要

■端子の概要



ピン	端子記号	内　容
1	UART TX	UART送信データ（LVTTLレベル）※1
2	UART RX	UART受信データ（LVTTLレベル）※1
3	I2C SCL	I2C通信クロック
4	I2C SDA	I2C通信データ
5	Mode1	モード設定ピン1
6	UART RTS	UART 送信要求（LVTTLレベル）※1
7	UART CTS	UART 送信可能（LVTTLレベル）※1
8	Mode2	モード設定ピン2
9	拡張端子	将来の拡張用です、何も接続しないでください
10	拡張端子	将来の拡張用です、何も接続しないでください
11	WAKE	Wake Upピン　※2
12	Vbat	内蔵RTC用電源入力 +3V～+3.3V
13	Vcc	+3.3V　本体電源　（±5%以内）
14	$\overline{\text{RESET}}$	ハードウエア本体リセット
15	GND	電源GND
16	拡張端子	将来の拡張用です、何も接続しないでください

※1:MSC-MOD60の電源電圧は3.3Vであり、ロジックレベルはLVTTLレベル(3.3V)ですが、すべてのI/OピンはTTLレベル(5V)のロジックもそのまま接続できます。(5Vトレラント機能搭載)

※2:WAKEピンは、MSC-MOD60のハイバーネートモードからの復帰時に使用するピンですが、扱いについては注意がありますので、詳しくは本書の"WAKEピンとハイバーネートモードの使用について"の項をお読み下さい。

※3:拡張用端子(9,10,16ピン)には何も接続せず必ずオープンとしてください。GNDやVccへの接続はしないでください。

電源について

MSC-MOD60の電源電圧は、DC3.3Vです。許容誤差は±5%以内です。電源ピン(13ピン)に印加された電源は、内部のCPU及びSDカードに直接給電されますので、品質のよい電源をご使用下さい。

電源は+3.3Vで100mA以上が取り出せる電源をVccピン(13ピン)及びVbatピン(12ピン)に接続します。GNDは、GNDピン(15ピン)に接続します。

Vbatピン(12ピン)は、内蔵のRTC用の電源ですが、RTCを使用しない場合でも、Vbatピンには+3V～+3.3Vの電圧が印加されている必要があります。実際の電源接続例は次の"リアルタイムクロックの使用について"の項をご覧下さい。

※3端子レギュレターを使用する場合の注意
3端子レギュレターで電圧を調整してMSC-MOD60の電源として使用する場合には、レギュレターの+3.3V側に発振が起こらないように十分注意して設計をしてください。パスコンの最適化や入力側に印加される電源の品質の向上などにご留意下さい。
リプルの重畳した電源の場合MSC-MOD60が誤作動を起こすことがあります。

リアルタイムクロックの使用について

MSC-MOD60には、本体にリアルタイムクロック(RTC)を搭載しています。RTCは、時間を計測してファイルやフォルダの作成時に、タイムスタンプを付けるために使用されます。"G F"コマンドで現在日時の取得ができます。

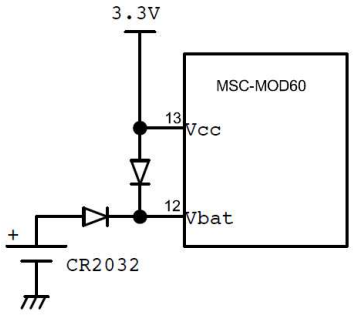
RTC機能を使用する場合には、別途付属の32.768KHzの水晶発振子を基板上のQ1ピンに半田付けしてください。この発振子がRTCのクロック源となります。

RTC機能は、Vccピン並びにVbatピンに電源が供給されている間、時計計測を行います。(事前に現在時刻の設定と、"T B"コマンドによ計測開始の処理が必要です。)

MSC-MOD60では、Vccピンの電源が切断されても、Vbatピンに+2.0V～3.3Vの電源を給電し続けることで、RTCは時間計測を続けることができます。CR2032などのボタン電池でバックアップすることができます。Vbatピンの消費電流は、+3.0V時に約560マイクロA程度です。Vbatピンの電源が切断されると、RTCの時計計測は停止します。RTC機能は、再度"S"コマンドを使用してRTCに日時を再設定しないと使えません。なお、一度RTCの設定がクリアーされてしまった場合には、RTCの保持する値はまったく意味のない日時になります。(※決まった値にはなりません。)よって、RTCがクリアーされた後のファイルに保存されるタイムスタンプは全く意味のない日時となります。また、"G F"コマンドで取得する日時の値も意味のない日時となります。

バックアップ機能を使用する場合には、下記のようにダイオードを2本使用して電源を切り替える簡易回路を作ります。

※ダイオードは1S1588互換品などが使えます。



RTCバックアップ用電源(CR2032)追加回路例

※Vbatピンは、RTCを使用しない場合でもMSC-MOD60を動作させるために+3V～3.3Vの電圧電圧を印加する必要があります。

シリアル通信について　～UART通信～

MSC-MOD60は非同期式シリアル通信と、同期式シリアル通信のI2Cに対応しており、モード設定ピン1(5ピン)と、モード設定ピン2(8ピン)で設定ができます。

	Mode1ピン	Mode2ピン
UARTモード	0	0
I2Cモード	0	1

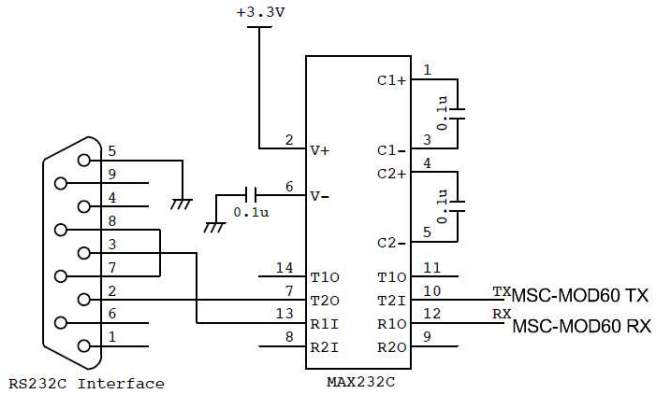
※0はGNDへ接続、1は+3.3Vへ接続

UART通信(非同期式シリアル通信)で使用する場合には、両ピンともGNDに接続してください。なおこの設定は、MSC-MOD60の電源投入前に設定し、モードピン設定後に電源を投入してください。

■UARTモードでの使用方法

非同期式シリアル通信(UART)通信を行う場合には、マイコンや、RS232Cポートを搭載したパソコンが必要です。なお、ロジックの電圧レベルはLVTTLレベル(3.3V)ですのでパソコンのRS232Cポートと接続する場合には、MAX232等のレベル変換ICを介して接続する必要があります。

・パソコンとの接続回路例



MSC-MOD60の通信条件はデフォルト設定では下記の通りです。

・通信速度:　9600bps
・データ長:　8ビット長
・パリティ:　なし
・ストップビット:　1

通信速度はシリアルコマンドで変更することができます。但し、設定した通信速度は電源を切断するとリセットされ、9600bpsに戻ります。通信速度を変更して使用する場合には必ず9600bpsで通信できる環境が必要となります。

■UART通信でのRTS/CTSハンドシェイク通信について

MSC-MOD60は、CTS/RTSハンドシェイク通信に対応しています。ハンドシェイク通信を使用することで信頼性の高いデータ通信を行うことができます。

CTSピンは、MSC-MOD60への送信可能を通知する入力ピンです。このピンがHレベルの時、MSC-MOD60はデータを送信せずLレベルになるのを待機します。ホスト側が処理をしていてデータ入力できない時に使用します。

RTSピンは、MSC-MOD60から送信要求を出力するピンです。このピンがHレベルの時は、MSC-MOD60のバッファはフルになっています。MSC-MOD60は、16バイト長のUART用バッファを持っています。ハンドシェイク通信を使用しない場合には、必ずCTSピンはLレベルに(GNDと接続)、RTSピンはオープンにしてください。

■UARTの通信速度について

通信速度は、9600bps～921600bpsまで8段階で設定できます。但し、パワーモードが省電力モードの時は9600bps～460800bpsまでの7段階となります。

デフォルト(工場出荷時設定)では、9600bpsに設定されています。通信速度の変更は、9600bpsで通信を行い、コマンドによって通信速度を設定します。よって、9600bps以外の速度で通信を行いたい場合でも設定変更のために、最低限1台のパソコンまたはマイコンは9600bpsで通信できなければなりません。なお設定した通信速度は電源を切断するとリセットされて9600bpsになります。

■PICマイコンとの接続について

PICマイコンなどLVTTLレベル又はTTLレベルのマイコンと接続する場合、MSC-MOD60のTX及びRXピンはマイコンに直結できます。MSC-MOD60の電源電圧は3.3Vであり、ロジックレベルはLVTTLレベル(3.3V)ですが、すべてのI/OピンはTTLレベル(5V)のロジックもそのまま接続できるようになっています(5Vトレラント)。よって、PICマイコンなど制御用マイコンがTTLレベルのデバイスであっても、そのまま直結できます。

シリアル通信について　～I2C通信～

I2C通信はフィリップスセミコンダクター社で開発された同期式シリアル通信方式で、正式には"アイスクエアードシー"と読みます。

MSC-MOD60は、同期式シリアル通信のI2C通信でも制御が可能です。I2Cモードにするには、MSC-MOD60のモードピンを設定して、電源を投入することでI2Cモードになります。I2Cモード時のMSC-MOD60はスレーブデバイスとして動作し、スレーブアドレスは0xA4に固定されています。アドレス値は変更できません。また、1つのバスに複数のMSC-MOD60を接続することはできません。

通信速度は100kbpsの標準モードと400kbpsの高速モードに対応しています。なお高速モード使用時はパワーモードをハイパワーモードで動作させておくことを推奨します。(省電力モードでは動作に不具合が生じることがあります。)

■MSC-MOD60をI2C通信モードに設定する

電源投入時に、Mode1ピン(5ピン)をLow、Mode2ピン(8ピン)をHighと設定することでI2C通信モードになります。設定は電源投入時にしか行えません。

■I2Cモード実行時のピンアサイン

I2Cモードにすると、ピン配置は下記のようにアサインされます。

ピン	端子記号	内　容
1	DATARDY	データレディ
2	BUSY	ビジー
3	I2C_SCL	I2C通信クロック
4	I2C_SDA	I2C通信データ

※その他のピンに変更はありません。

使用するピンは、クロックであるI2C_SCL(3ピン)と、データ線であるI2C_SDA(4ピン)です。また、必要に応じて、DATARDYピン(1ピン)と、BUSYピン(2ピン)を使用します。

I2C_SCLと、I2C_SDAは共に4.7KΩ程度の抵抗でプルアップする必要があります。

DATARDYピンと、BUSYピンはMSC-MOD60の出力ピンで、DATARDYピンがHレベルの時、MSC-MOD60側からホスト側に対して送信するデータがあることを示します。(MSC-MOD60はデータは受信しません。)よってホスト側は、なるべく早くデータを読み込む必要があります。

BUSYピンがHの時は、MSC-MOD60の内部入力バッファがいっぱいであることを示しますので、この場合にはホストはデータを送信できません。(BUSYピンがHの時以外は、いつでもデータと、コマンドはMSC-MOD60に送信できます。)
よって、いずれのピンもLレベルである時に、MSC-MOD60はスレーブデバイスとしてデータを受信することになります。

■I2C通信の手順

電源投入直後又はリセット直後は、MSC-MOD60は、下記の文字列を送信します。

```
 GHI Electronics, LLC
----------------------
    Boot Loader 2.05
    uALFAT(TM) 3.13
!00
```

"GHI･･･"の前には1つキャリッジリターン(CR=0x0D)と、Gの前にはスペース(0x20)が入っています。また各行はCR(0x0D)で終端されています。これらの文字列を送信するため、MSC-MOD60のDATARDYピンはHとなります。最初は、まずこの文字列を受信しなければなりません。通信手順は、一般的なI2C通信の手順と同じですが、下記にホスト側のC言語で作った例を示します。

最初にスタートコンディションを発行し、その後スレーブアドレス0xA4と、マスター側が受信しますので、コントロールバイトを1として読み込み設定にしてMSC-MOD60に送信します。(MSC-MOD60のスレーブアドレスは、7ビットです)

続いて、DATARDYピンがLレベルになるまで、受信を繰り返します。Lレベルになった時点でホスト側の受信処理は終了します。

```
void main(){

    unsigned short rec;

    while(DATARDY){
        I2C_Start();
        I2C_Wr(0xA4+1);
        rec = I2C_Rd(0u);
        I2C_Stop();
    }
}
```

続いて、ホスト側からコマンドを送信する例を紹介します。例としてホスト側からVコマンドを送信して、バージョン情報を取得する方法を紹介します。手順は、一般的なI2C通信と同じです。

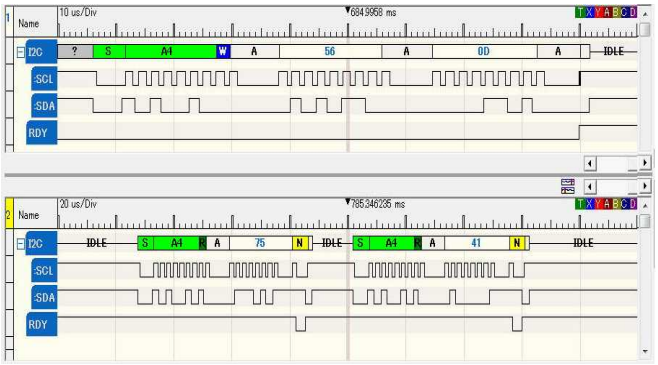
```
void main(){

    I2C_Start();
     I2C_Wr(0xA4+0);
     I2C_Wr('V');
     I2C_Wr(0x0D);
     I2C_Stop();
     delay_ms(100);

}
```

上記例はホスト側からの送信部分だけです。受信は、先ほどの例と同じ方法で受信します。マスター側から送信しますので、コントロールバイトは0xA4の7ビットスレーブアドレスと、0のWモードとなります。
続いて、Vコマンド(0x56)を送信し、最後にCR(0x0D)を送信して通信を完了します。CR送信直後に、DATARDYピンがHとなり、MSC-MOD60側が "uALFAT3.13"　というようなバージョン情報を返す準備がととのったことを通知します。

先の手順と同じようにしてDATARDYがLになるまで、受信を繰り返してデータを受信してください。
本例をロジックアナライザで観察すると下記のようになります。



上段がホスト側→MSC-MOD60へのコマンド送信。
下段が、MSC-MOD60側→ホストへのレスポンスです。(2バイト分)

なお、本体の状態がリセット直後の不安定な場合や、DATARDYピンがHレベルでない時(送信するデータがない時)に、ホスト側が送信を要求すると、MSC-MOD60は0xFFを返します。(通常は先のプログラム例の通りDATARDYピンの状態を判定して、Hの時のみ受信しますので、本マニュアルに記載の内容で通信すればこの状態は起こりません。)半二重通信を確実に行うため、DATARDYピンの状態は常に監視する必要があります。

その他のコマンド体系等は、UARTとI2Cとは共通です。

MSC-MOD60のハードウエア設計について

■電源回路について

MSC-MOD60の電源電圧は+3.3Vです。消費電流は最大で40mA程度となりますので、余裕を考えて100mA以上を取り出せる電源回路をご準備下さい。必ず安定化されたものをご使用下さい。

+3.3Vの電源は、13ピンのVccピンと12ピンのVbatピンの両方に給電する必要があります。VbatピンはRTCの時間計測のバックアップ用電源端子にもなります。

■モードピン(5ピン、8ピン)の設定

電源投入の前に、モードピンの設定を行って下さい。モードピンの設定は電源投入中は行わないでください。(電源供給中の設定の変更は反映されません。)

モードピンにてシリアル通信モードをUART又はI2Cのどちらかに設定します。

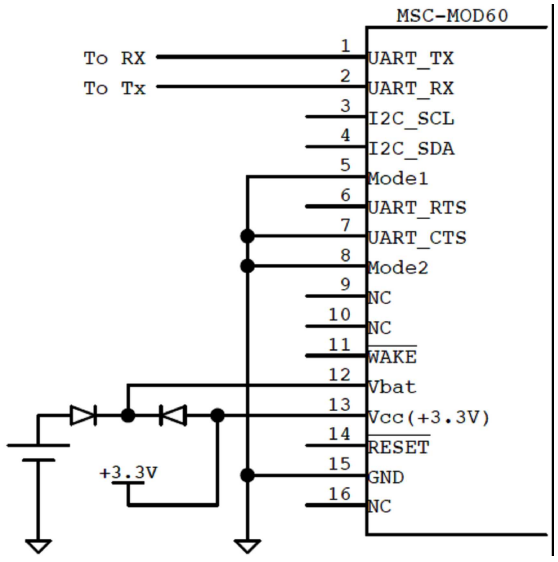
	Mode1ピン	Mode2ピン
UARTモード	0	0
I2Cモード	0	1

※0はGNDへ接続、1は+3.3Vへ接続

■最も基本的な配線例

下図はRTCのバックアップ用のバッテリを使用した場合の基本的な配線例です。シリアル通信モードはUARTとなっています。

ダイオードを2本用いて、Vcc切断時にVbatピンに接続されたバッテリーでRTCの時計計測が継続できるような簡易回路を作っています。



基本回路例(参考)

■NCピンの取扱について

NCピン(9,10,16)は将来の拡張用のピンとして設けられています。使用しませんので必ずオープンにしてください。GNDやVcc、その他の信号線などには接続しないようご注意下さい。

■リセットピンの取扱について

14ピンのリセットピンは、本体内部で10KΩの抵抗によってプルアップされています。本体のハードウエアリセットをかける場合には、このピンをLレベルにします。使用しない場合には、オープンにしておきます。(リセットをかける時のLパルスは最低30mS以上です)

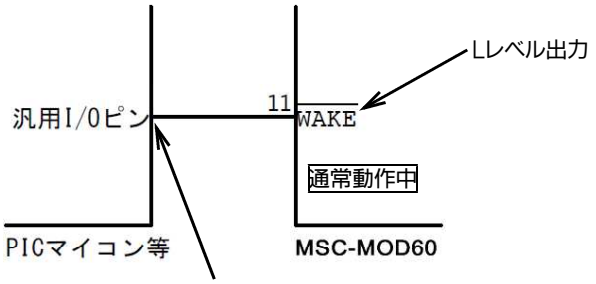
■WAKEピンとハイバネートモードの使用について

WAKEピン(11ピン)は、本体を"Z H"コマンドでハイバネートモードに移行させた後、通常の動作モードに復帰させる時に使用ためのピンですが、使用に際してはいくつかの条件があります。

このピンをハイバネートモードからの復帰させるためのWAKE機能として使用する場合、このピンはMSC-MOD60への入力ピンですが、ハイバネートモードからのWAKE実行時以外は、このピンはオープンにしておく必要があります。プルアップなどの回路は外部に取り付けないでください。

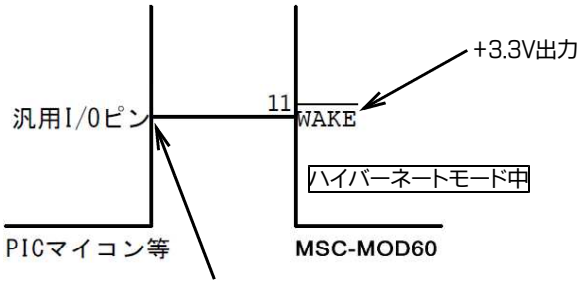
外部にマイコンなどがあり、そのマイコンからWAKE動作をさせたい場合には、次のように設計する必要があります。

①MSC-MOD60の通常動作時(ハイバーネートモード以外の時)は、このピンに接続されているマイコンのピンは入力ピンに設定します。(Hレベルや Lレベルの信号はマイコンからは出力しないでください)この時、MSC-MOD60のWAKEピンはLレベルになっており、出力ピンの設定になっています。



マイコン側は入力ピンに設定

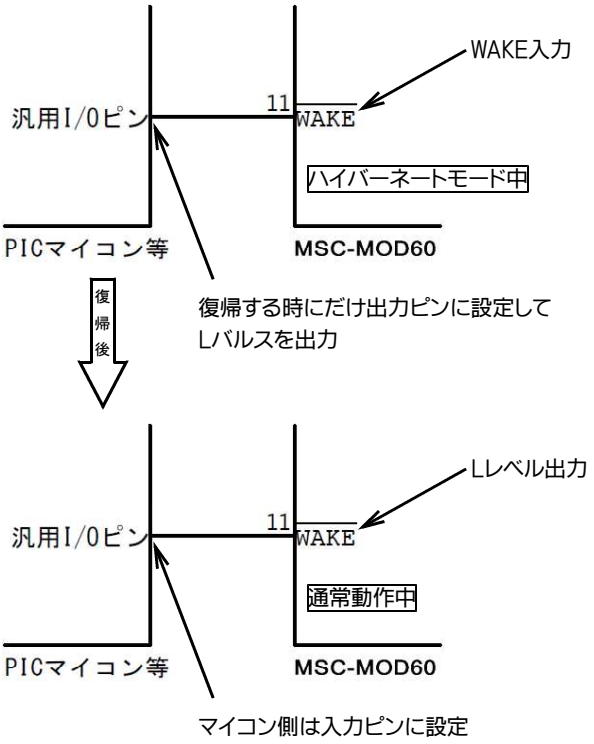
②MSC-MOD60がハイバーネートモードに移行すると、WAKEピンには、+3.3Vの電圧が現れます。マイコン側のピンは入力設定を持続させます。(ハイバネートモード中はこのWAKEピンがHレベルなることにご注意下さい。)



マイコン側はハイバネートモード中は
入力ピン設定を維持する

※WAKEピンは、ハイバネートモード中にHレベルとなり続けますので、マイコンへ通知するインジケーターとして使用もできます。(電流は1マイクロA未満程度しか取り出せませんのでご注意下さい。)

③ハイバーネートモードから通常モードへ復帰させたい場合には、マイコン側のピンを出力設定に変更し、Lレベルをマイコン側から出力してWAKEピンに印加します。(Lレベルのパルスは30ミリ秒以上です)
通常モードへ復帰すると、再度WAKEピンはLレベルになりますので、マイコン側のピン設定を入力設定に切り替えます。



※なおWAKEピンは、$\overline{\text{RESET}}$ピンがLレベルになっている時も+3.3Vが出力されます。

※PICマイコンからMSC-MOD60の制御を行う場合で、WAKEピンの制御もマイコン側から行う場合にはWAKEピンと接続したピンのポートのTRISxレジスタを操作することで入力方向・出力方向の設定ができます。

MSC-MOD60の準備

MSC-MOD60を使用する際には、以下の手順でハードウエアの準備を行ってください。

1 MSC-MOD60本体のカードスロットに、SDカードを挿入します。
"カチッ"と音がするまで挿入します。

※MSC-MOD60は、FAT16及びFAT32のファイルシステムに対応しています。使用するSDカードは必ず上記のファイルシステムにてフォーマットしておいてください。

2 MSC-MOD60に電源を投入します。
※電源接続前にシリアル通信のモード設定が正しく設定されているか確認ください。電源投入直後、下記のような文字列がMSC-MOD60から送信されます。

```
 GHI Electronics, LLC
 ----------------------
    Boot Loader 2.05
    uALFAT(TM) 3.13
!OO
```

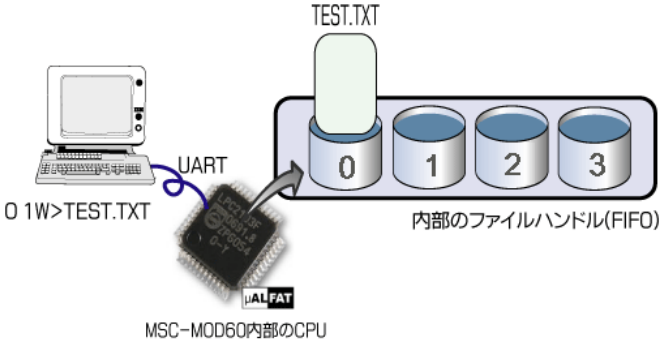
これは、内部で使用しているデバイスのファームウエアバージョンを通知するもので、電源投入直後とリセット直後に送信されます。

3 UART通信ができるデバイス又は、RS232Cレベル変換ICを介してパソコンと接続します。
UARTで使用するピンは、UART TXピン(1ピン)、UART RX(ピン)及びGNDピン(15ピン)です。またハンドシェイク通信を行わない場合には、CTSピン(7ピン)をGNDに接続してください。

4 これでハードウエアの準備が完了しました。

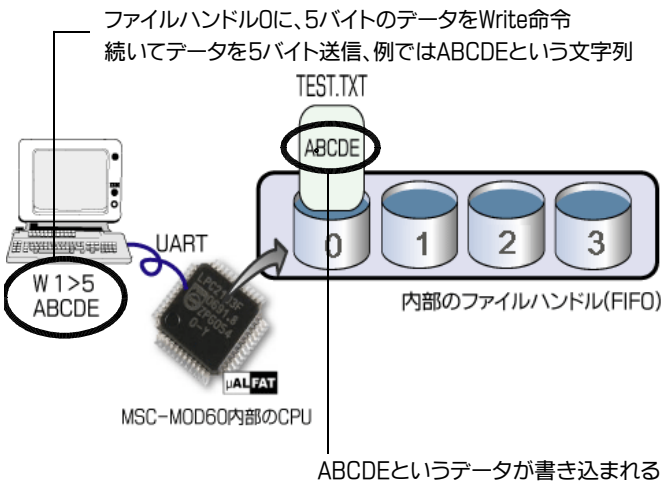
MSC-MOD60の基本的なファイル操作の仕組み

MSC-MOD60はファイル単位で操作します。
ファイルはそのまま扱うのではなく、一度MSC-MOD60の内部にある"ファイルハンドル"と呼ぶ、バッファにファイルを開きます。
ファイルハンドルは4つ(0~3)あり、操作をしたい(データを読んだり、データを追記するような各種ファイル操作)ファイルをOコマンドでファイルハンドルに展開してから各種操作を行います。
Oコマンドは、指定したファイル名のファイルを、指定したファイルハンドルに展開するコマンドです。以後、そのファイルへのアクセスはファイル名ではなく、ファイルハンドルの番号で行うことになります。



上図は概念図ですが、MSC-MOD60内部にはFIFOがあり4つのファイルハンドルがあります。

0コマンドでファイルハンドル1にTEST.TXTというファイルをWモードで展開すると、TEST.TXTというファイルがファイルハンドル1に作られます。以後は、ファイル名の"TEST.TXT"ではなく、ファイルハンドル1に対して各種操作をすることで、このファイルを自由に扱うことができるようになります。



なお、最後にファイルハンドルをクローズする際に、実際にメモリーデバイス上にファイルが作られますので、ファイル操作を終了する場合には必ずクローズコマンドでファイルハンドルを閉じて終了します。この時点で初めてSDカード内にファイルの実体が作られます。ファイルハンドルに開いたファイルはいつでも、閉じることができます。

ファイルハンドルを閉じる操作を行う前にSDカードをMSC-MOD60から外してしまったり、MSC-MOD60の電源を切断してしまうと、編集データや新規作成したファイルはすべて記録されず消えてしまいます。

■SDカードへの書き込みの速度

MSC-MOD60のマイクロSDカードへのデータ書き込みの実効速度は、約42kBytes/s程度(約336kbps)です。1バイトあたり23マイクロ秒程度の書き込み速度です。この値は、おおよその実効値であり、カードの種類や容量などによって多少前後することがあります。

なおSDカード及びFATファイルシステムの特徴として1セクターは512バイトとなっています。そのため仮に1バイトのデータを書き込む場合であっても512バイトを書き込まなくてはなりません。よって書き込むデータサイズが増えてもそれに比例して書き込み時間が長くなる、又はその逆で書き込むデータサイズを小さくすることでそれに伴い書き込み時間が短くなるかというと単純にそうはいきません。また、データの信頼性を確保するためCRC16が採用されており、冗長コードを付加してエラーをなくしています。これらの演算にも時間がかかっています。

なお書き込み時間や読み込み時間はSDカードに採用されているメモリーICやコントローラICに依存的で、実効時間はカードによってばらつきがあります。そのため平均速度は約42KB/sですが実際には使用するメモリーカードによってばらつきがあることをご了承下さい。

MSC-MOD60の扱えるファイル名とディレクトリ名

MSC-MOD60の扱えるファイル名とディレクトリ名には下記のような決まりがあります。

■ファイル名とディレクトリ名についての制限

○MSC-MOD60では、1バイト文字の半角英数のみファイル名及びディレクトリ名として使用できます。全角などの2バイト文字並びに日本語は使用できません。
○ファイル名ディレクトリ名ともにスペースは使用できません。
○ドット(.)は拡張子を示すために用いることができますが、ファイル名には使用できません。
○次の記号は使用できます。
$ % ' - _ @ ! ( ) { } # &

■ファイル名とディレクトリ名の長さについて

MSC-MOD60では、ロングファイルネーム規則に対応しています。本製品はマイクロソフト社のLFN規則のライセンスを受けており、長いファイル名が使用できます。

コマンドフォーマットとACK記号

■MSC-MOD60のコマンドフォーマットとキャリッジリターン

MSC-MOD60のコマンドは1バイトで、必要に応じて後ろにスペース(0x20)や、引数を含みます。

各コマンドの最後は、必ずキャリッジリターン(0x0D)を送信して終端します。MSC-MOD60は、このキャリッジリターンを受信した時にデータを処理します。

例えば、本体のファームウエアのバージョンを取得するコマンドにVコマンドがあります。Vは、16進数では0x56です。よって、MSC-MOD60に送信する場合には、 0x56 0x0D を送信することになります。
また、引数(パラメーター)を指定するコマンドの場合には、コマンドの後にシングルスペース(16進数では0x20)を挟みます。
まとめると次のようなフォーマットになります。

C{sp}Parameter1{sp}Parameter2{sp}.....{cr}

・C はコマンドです。
・{sp}はシングルスペースで、0x20です。
・Parameterは引数で、必要な場合に挿入します。
・{cr}はキャリッジリターンで、0x0Dです。

■MSC-MOD60の戻り値

MSC-MOD60にコマンドを送信すると、その内容に応じた戻り値が返ります。MSC-MOD60では、コマンドを正しく受信し処理が正常に行われると、"!00"(16進数では、0x21 0x30 0x30 0x0D)が返ります。この !00 はACK(ACKnowledgement)で、データ転送が正常に終了した時に出力されます。

これ以外の戻り値で16進値の前に ! (0x21)が付いているものは、何らかのメッセージ又はエラーメッセージとなります。なお、コマンドにもよりますが、多くのコマンドで共通することは、1つのコマンド送信後は必ず !00 のACKコマンドを受信したことを確認してから次のコマンドを送信しなければなりません。
※!00の後ろにはCRがありACKは4バイトです。

■MSC-MOD60が扱う数値

MSC-MOD60では、標準の数値表現が16進数です。記号などを付けなくてもすべての数値は16進数となります。

ファームウエアの更新について

MSC-MOD60では、ファイルシステムの制御やメモリカードの制御、その他シリアルインターフェイスを米GHI社製のCPUが行っています。このチップはuALFAT(マイクロアルファット)と呼ばれるチップで、現在このCPUに書き込まれているファームウエアバージョンは3.12となっています。ファームウエアのバージョンは、Vコマンドで確認できます。

ファームウエアの更新は、最新版ファームウエアをSDカードに保存した後、UART通信又はI2C通信経由で、"X S"コマンドを送信することで、行うことができます。新しいファームウエアがリリースされた場合には、当方のWEBサイトにて告知致します。またダウンロードが可能となりますので、下記の方法でファームウエアを最新版に更新してご使用頂くことができます。

本製品を組み込んだシステムを開発される際には、今後のファームウエアアップデートが可能なようにシステムの設計並びにソフトウエアの開発をして頂けますようお願い致します。

1 SDカードを用意します。FAT16又はSDフォーマッタを使用してSDカードをフォーマットしてください。

2 当方のWEBサイトから最新版のファームウエアをダウンロードします。ファームウエアは下記のサイトからダウンロードが可能です。
現在お使いのバージョンよりも新しいバージョンがリリースされているかどうかご確認ください。
http://www.microtechnica.net/manual/

3 ファームウエアは、"UALFATFW.GHI"というファイル名です。
このファイルを1で用意したSDカードにコピーしてください。

4 3のSDカードをMSC-MOD60に挿入します。
※MSC-MOD60が使える状態である必要があります。

5 シリアル通信(UART又はI2C)にて、下記のコマンドを送信します。

X△S{cr}

※△は半角スペース(0x20)、{cr}はキャリッジリターン(0x0D)です。X(0x58)とS(0x53)はともに大文字で、XとSの間に1つ半角スペース(0x20)を入れてください。

※SDカードを初期化するIコマンドは必要ありません。

6 ファームウエアのアップデートが開始されます。アップデート中は、"Wxx{cr}"という(xxは文字列)文字列が不定長で返ります。アップデート中はこの文字列をすべて受信してください。
I2C通信の場合や、UART通信でもハンドシェイク通信をしている場合には、文字列を遅滞なくMSC-MOD60側から受信してください。
アップデートは約3秒程度で終わります。

7 アップデートが終了すると、電源投入時又はリセット時に出力される下記の文字列が出力されます。
ファームウエアバージョンが更新されているかご確認ください。
なおファームウエアバージョンはVコマンドでも確認できます。

```
 GHI Electronics, LLC
----------------------
    Boot Loader 2.05
    uALFAT(TM) x.xx
!00                  └─ 更新後のバージョンが表示されます。
```

8 MSC-MOD60を再起動します。

ファームウエアのアップデートにより、新しい機能が追加されたり、既知の不具合や問題が解決されたりすることがありますので、ファームウエアのアップデートができるように、機器の設計をして頂けますようお願いいたします。また、ファームウエアは常に最新のものをご使用ください。

とりあえず使ってみましょう

MSC-MOD60には、さまざまな機能が搭載されており、シリアルコマンドも豊富です。しかし、ここでは、とりあえず動作させて使ってみることで、MSC-MOD60の最も基本的な使い方を知って頂くことにしましょう。詳しいシリアルコマンドについては次の項から記載がありますので、そちらでお読みください。

なお、本項ではMSC-MOD60から返るコマンドは、斜体で記載します。また、コマンド内の<cr>はキャリッジリターン(0x0D)、<sp>は半角スペース(0x20)を示します。MSC-MOD60からの戻り値は、すべてCRで終端されているとして、<cr>は記載しておりません。

■SDカードに新規にファイルを作成する

まず基本的な操作であるファイルの新規作成を体験しましょう。1GB程度のSDカードをご用意ください。

1 SDカードをFAT32でフォーマットしましょう。
SDカードをパソコンに接続して、FAT32でフォーマットしてください。(FAT16でも使用できます)

2 MSC-MOD60の配線を行いましょう。今回はパソコンのRS232Cポートと接続して実験することにします。シリアルインターフェイスはUART(ハンドシェイクは使用しません)を用います。
VccピンとVbatピンに3.3Vが、印加されているか確認しましょう。
その他配線が正しく行われていることを確認しましょう。パソコンと接続する場合には、MAX232等のレベル変換ICを介してパソコンのRS232Cポートと接続する必要があります。またモードピンはUARTモードに設定しておく必要があります。

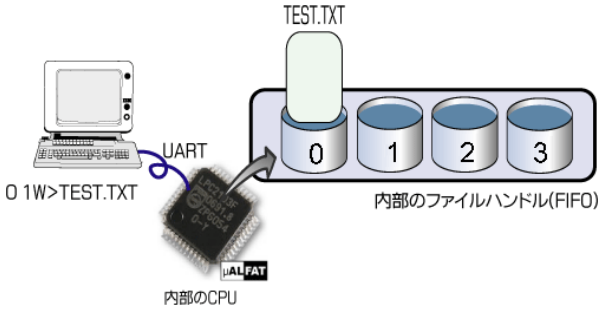
3 MSC-MOD60にSDカードを装着します。

4 シリアル通信で、"I"コマンド(アイ)を送信します。

```
I<cr>
!00
```

"I"は、ASCIIコードで0x49です。コマンドの最後は必ずキャリッジリターンで終端します。キャリッジリターンは0x0Dです。
SDカードが正しく初期化できると、!00のACKがMSC-MOD60から返ります。

5 ファイルを新規に作成しましょう。MSC-MOD60では、すべての操作はファイルハンドルというCPU内のメモリー空間に一時的にファイル作成し、そのファイルハンドルに対してデータの書き込みなどの操作を行うことになります。



ファイルハンドルは0~3までの4つが利用できます。
ここでは、ファイルハンドル0に、例として"TEST.TXT"というファイルを作成することにしましょう。

6 コマンドを次のように送ります。

```
O<sp>0W>TEST.TXT<cr>
!00
```

"O"(オー)コマンドはファイルハンドルにファイルを開くコマンドです。
"0"(ゼロ)はファイルハンドルの番号、"W"はWモード、すなわちファイルに対してデータを書き込むモードという意味です。
そして、">"(0x3E)に続いて、ファイル名を文字列として送信します。

なお、最初の"O"の後ろにはスペース(0x20)が1つ入りますので注意してください。
正しく受信されると、ACKの!00が返ります。

7 では実際にデータを送信して、TEST.TXTファイルらデータを書き込んでみましょう。データはどんな形式でもかまいません。バイナリ形式でもいいですし、ASCIIコードで送れば文字列となります。
ここでは、文字列として "HELLO WORLD" という文字列を送信してファイルに書き込むことにしましょう。

"HELLO WORLD"は文字数とすると11文字です。よってデータサイズは11バイトとなります。書き込むデータのサイズを指定するにはWコマンドを使用します。次のようにコマンドを送信しましょう。

```
W<sp>0>B<cr>
!00
```

MSC-MOD60では、すべての値は16進数として表現します。11バイトは、16進数で0x0Bバイトですので、ここでは"B"を指定します。
Wコマンドは指定したファイルハンドルに対して何バイトのデータを書き込む‥ということを指定するコマンドです。

なお、最初の"W"の後ろにはスペース(0x20)が1つ入りますので注意してください。
正しく受信されると、ACKの!00が返ります。

8 この状態で、MSC-MOD60は11バイトのデータの受信待機状態となります。文字列、"HELLO WORLD"を送信してみましょう。

```
HELLO WORLD
$0000000B
!00
```

書き込むデータを指定したデータサイズ分送信し終わると、自動的にMSC-MOD60から、書き込みが完了した旨の通知と、ACKが返ります。
"*$0000000B*"は、11バイトを確かに受信しました、という通知です。その次の!00は書き込み完了のACKです。

なお、データの最後にキャリッジリターンが付いていると、それに対応した!00が1回返ります。

9 ここまでの作業でデータはSDカードのメモリー領域に書き込まれました。しかしFATファイルシステムでは、データを書き込んだだけではそのデータを使うことはできません。
データの存在する位置を管理領域というテーブルに書き込んで初めて「使えるデータ」となります。

MSC-MOD60では、この作業を行うためにクローズコマンドの"C"コマンドを使います。Cコマンドを使用すると、指定したファイルハンドルが閉じられ、データが参照できるようになります。

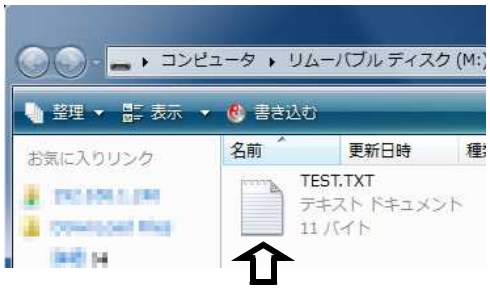
コマンドを次のように送ります。

```
C<sp>0<cr>
!00
```

ファイルハンドル0を閉じるというコマンドです。
ACKの!00が返った時点で、SDカードには"TEST.TXT"というファイルが作成されます。

※このCコマンドを送信しないで、SDカードを抜いてしまったり、MSC-MOD10の電源を切断してしまったりすると、データはすべて消えてしまいますので十分ご注意ください。

10 ではこのSDカードをパソコンに装着して、正しくファイルが作成できているか確認しましょう。



ファイルが作られていることが確認できます。メモ帳で開いてみましょう。

正しく"HELLO WORLD"という文字列が作成されています。

このようにMSC-MOD60ではシリアルコマンドで簡単にファイルを作成することができます。もちろんテキストファイルだけでなく、形式を問わず作成できます。
カンマ区切りのCSVファイルや、JPEG・MP3などといったバイナリファイルももちろん作成できます。

ここでは簡単なファイル作成例を紹介しましたが、様々なシリアルコマンドを使うことで、さらに複雑な操作ができます。

続いてファイルからデータを読み出す方法も紹介しましょう。

■SDカード内のテキストファイルからデータを読み込む

基本的な操作である、SDカード内のテキストファイルからデータを読み出す操作を体験しましょう。1GB程度のSDカードをご用意ください。

1 データを読み出すためにあらかじめ、SDカード内にファイルを保存しておく必要があります。
"TEST.TXT"という名前のファイルに次のような文字列を記述してSDカードのルートディレクトリに保存しておきます。

2 先のファイル作成の例と同じで、まずSDカードをMSC-MOD60に装着し、"I"コマンドで初期化を行います。

```
I<cr>
!00
```

3 今回は、データを読み込むモード(Rモード)で、ファイルハンドルにファイルを展開しますので、次のようにコマンドを送信します。

```
O<sp>0R>TEST.TXT<cr>
!00
```

ACKが返ったことを確認します。

4 読み出しは"R"コマンドで行います。
読み出すデータのサイズを指定します。指定は16進数表現で行います。ここでは試しに16バイト(0x10)を指定してみましょう。

```
R<sp>0Z>10<cr>
```

0の後ろの"Z"(ゼット)は、超過文字といって、ファイルハンドル内にあるファイルのデータサイズが、指定したデータサイズよりも小さい場合、超過分のデータをこの文字で埋めるというものです。
データサイズ内であれば、特に気にする必要はありません。

さて、上記のコマンドを送信すると、下記のようにデータが返ります。

```
!00
HELLO WORLD
Thi$00000010
!00
```

最初の!00はACKで、その後から実際のデータが返ります。データの最後には$マークに続き、読み取ったデータサイズ、その次に再度読み取り完了のACK、!00が返ります。

今回は16バイトを指定しました。しかし14バイトしか返ってきていないように見受けられます。
しかし、これは間違えではありません。なぜならば、文字列"HELLO WORLD"と"This is a test file"の文字列の間には改行が入っているからです。

Windowsシステムでは、改行はCR(0x0D)と、ラインフィード(0x0A)の2バイトから構成されます。よって、Windowsでテキストデータを作った場合、改行の部分は2バイトのデータとして扱われます。

先ほどのTEST.TXTをバイナリエディタで見てみましょう。

改行部分に0x0Dと0x0Aが含まれていることが分かります。
改行はテキストデータとして扱っていると意識しないことが多いので、おかしいと思ってしまいがちなので注意が必要です。

5 さて、続いて残りの部分も読み込んでみましょう。
今度はテストとして再度20バイトを指定してみましょう。

```
R<sp>0Z>14<cr>
```

返ってきた文字列は下記のようになりました。

```
!00
s is a test file.ZZZ$00000011
!00
```

Rコマンドでは、ポインタの位置をリセットしたり、任意の位置にセットしない場合、前回の続きの位置からデータを読み出します。
今回は20バイトを指定しました。実際にはデータは17バイトしかありませんでしたので超過した分の3バイトの部分に、超過文字"Z"が付加されていることがわかります。

このようにMSC-MOD60では簡単にデータを読み出すことができます。データの読み出し位置(アドレス)は任意の部分に指定できる他、ルートディレクトリ以外に保存してあるデータももちろん読み出すことができますので、詳しくは各コマンドの解説をお読みください。

シリアルコマンド

■コマンドについての一般的な規則

MSC−MOD60を制御するためにはシリアルコマンドの授受をする必要がありますが、次の項目を必ず守るようにシステムを設計してください。

・いずれのコマンドでもデータサイズが200バイトを超えないようにしてください。

・各コマンドは必ずキャリッジリターン(CR=0x0D)で終端してください。MSC−MOD60ではCRを受信して初めてコマンドを受け付けます。

・ユーザーは必ずMSC−MOD60から戻る戻り値を確認するようにシステムを設計しなくてはなりません。各コマンド送信後に戻る戻り値は必ずチェックできるように設計し、コマンドが正しく実行されたかということを確認できるような設計をお願い致します。エラーが返った場合には、そのエラーに応じて適切に対応ができるような仕組みを作る必要があります。

・MSC−MOD60で扱うすべての番号はASCIIで表記された16進数の値です。例えば、10進数で数値10と指定したい場合には、MSC−MOD60には文字"A"(0x41)を送信しなければなりません。誤って、10進数表現のまま10として送信してしまうと、MSC−MOD60は、16進数の"イチゼロ"として処理されるため、実際には10進数で16として処理されてしまいます。

・MSC−MOD60が正しくコマンドを受信したり、受信したコマンドの処理を開始又は完了すると、多くの場合、ACKとして!00{cr}が返ります。エラーが発生した場合には、!xx{cr}としてエラー内容に対応したエラーが返ります。コマンドによっては1つだけでなくいくつかのエラーコードを返す場合もあります。

・以降のMSC−MOD60の制御コマンドを解説では、下記のような決まりで記述しています。

- *{cr}* はキャリッジリターン(0x0D)を表します。 ※戻り値の例ではCRは省略しています。行が変わる部分にはCRが付加されています。
- △印 はシングルスペース(0x20)を表します。
- 下記表記の中で斜体文字は、MSC−MOD60からの戻り値を表します。

■コマンド一覧表

コマンド	詳細	コマンド	詳細
V	ファームウエアバージョンの取得	#	エコー設定
Z	パワーモード設定	T	タイマーの初期化
S	現在日時の設定	G	現在日時の取得
B	通信速度設定	I	MMC/SDカードの初期化
K	メディアの空き容量、全容量の取得	@	ディレクトリリストの初期化
A	ディレクトリの移動	L	次のディレクトリエントリーの取得
M	ディレクトリの作成	R	ファイルの読み込み
O	ファイルオープン	P	ファイルシーク
W	ファイルに書き込み	D	ファイルの削除
Y	現在のファイル内のカレントインデックス位置の取得	?	ファイル又はフォルダーの存在の検索
E	フォルダーの削除	X	ファームウエアのアップデート
Q	クイックフォーマット	～	LFN規則ライセンス取得の有無確認

■各コマンドーの詳細

【コマンド】 **V***{cr}*

【動作】 MSC−MOD60のバージョン情報を取得する

【解説】 MSC−MOD60のファームウエアのバージョン情報を取得します。

【戻り値】 *uALFAT 3.13*
!00

【コマンド】 **#** *△ n{cr}*

【動作】 エコーバックの有効/無効を設定する

【引数】 n n=0の時はエコーバックは無効、n=1の時はエコーバック有効

【解説】 エコーバックとは、MSC-MOD60が受信したデータをそのまま送信元へ返す処理です。
デフォルト設定では無効になっています。

【コマンド】 **Z** *△mode>vvv{cr}*

【動作】 パワーモードを変更しUARTの通信速度を設定する(通信速度の設定変更のみの場合にはBコマンドを使用します)

【引数】 *mode* F・・・フルパワーモード(クロック周波数70MHzモード)
R・・・省電力モード(クロック周波数10MHzモード)
H・・・ハイバネートモード

vvv UARTの通信速度を下記の表を参考に指定います。ハイバネートモードを実行する場合には、指定しません。

【戻り値】 *!00* !00が2つ返ります。最初の!00は設定変更前の通信速度で返ります。
!00 2つ目の!00は設定更新後の通信速度で返ります。

【解説】 MSC-MOD60には、3つのパワーモードが搭載されています。本コマンドはパワーモードを変更するためのコマンドです。
UARTの通信速度の変更だけをする場合には、Bコマンドを使用します。

フルパワーモードは、内部のクロック周波数を70MHzで発振させて高速動作を行います。通常の動作の他、本体のファームウエアを書き換える場合にはこのモードで動作させます。フルパワーモードで動作時の平均消費電流は約39mAです。内部の処理が早いので、各種コマンドの処理などが高速に行えます。工場出荷時にはこのモードになっています。

省電力モードでは、クロック周波数を10MHzにして消費電流を抑えます。通常動作の時に使用できるモードです。このモードでの動作時の平均消費電流は約8mAです。動作クロックを落として消費電流を抑えていますので、コマンドによっては処理に時間がかかることがあります。また通信速度に制限があります。その他、ハイバネートモードへは直接移行できません。

ハイバネートモードは、MSC-MOD60を使用しない際に使用できる一時休止モードです。ハイバネートモードに設定すると消費電流は、1mA程度まで下がりシステムは最低限必要な部分を残して停止します。この状態ではシリアル通信のコマンド等は一切受け付けません。ハイバネートモードは、省電力モード実行時には実行しないようにしてください。復帰時にCPUのクロックモードが適切に設定されないため通信速度の設定にずれが生じることがあります。ハイバネートモードを実行する場合には、必ずフルパワーモードから実行するようにします。省電力モード時には一度、フルパワーモードへ移行後にハイバネートモードを実行するようにします。
ハイバネートモードから復帰する場合には、WAKEピンをLowレベル(Lレベルの30ミリ秒程度のワンショットパルス)にします。
500ミリ秒以内でハイバネートモードから通常モード(フルパワーモード)へ復帰します。復帰が完了すると、!00のACKが返りますのでこのACKを確認してからコマンドを送るようにします。ハイバネートモードへ移行してもWAKEピンによって通常モードへ戻れば、ハイバネートモード実行前に設定した各種設定や、ファイルハンドルの内容などは保持されています。なおMSC-MOD60をシステムリセットするとハイバネートモードは解除されます。

通信速度(bps)	フルパワーモード時	省電力モード時
9600	DCEF◎	1FAB
19200	6EEF	0C7C
38400	37EF	067C
57600	43F2	08E5
115200	1EF4	04E5
230400	0FF4	×
460800	05A9	×
921600	028B	×

◎印は、工場出荷時のデフォルト設定値です。 ×印はサポートされていません。

【使用例】 ①フルパワーモードで、通信速度115200bpsに設定する場合

Z F>1EF4*{cr}*

!00 設定前の通信速度でACKが返ります。

!00 新規に設定した通信速度115.2kbpsの通信速度でACKが返ります。

【補足】 本設定は、電源を切断すると初期設定のフルパワーモード、9600bpsに戻ります。設定を記憶することはできません。
ハイバネートモードは、必ずフルパワーモードから実行してください。省電力モードから実行することはできません。省電力モードからハイバネートモードへ移行したい場合には一度フルパワーモードへ移行後にハイバネートモードへ移行するよう設計してください。
本コマンドはパワーモードを変更するためのコマンドです。UARTの通信速度を変更するだけの場合にはBコマンドをご使用下さい。

T *△* **B** *{cr}*

【コマンド】

このコマンド送信後から時計計測を開始する

【動作】

CPU内のRTC機能の動作を開始させます。RTCは時計計測機能で、ファイルを作成する際などにタイムスタンプとして使用されます。

【解説】 RTC機能を使用するためには、MSC-MOD60のQ1に32.768kHzの水晶発振子が取り付けられている必要があります。

RTC機能を使用するためには、最初にT△Bコマンドを送信後、現在日時設定コマンド(Sコマンド)を送信します。
Sコマンドで現在日時を受信した段階から、現在日時に応じた時計計測が開始されます。

本コマンドは、RTC機能を動作させるためのコマンドです。本体の電源切断後でもVbatピンにバックアップ用の電圧が印加されていれば
【補足】 時計計測は保持されますが、Vbatピンへの給電もなくなると、時計計測は停止し、RTCの値は意味を持たない値となります。
その場合には再度、T△Bコマンド→Sコマンドで現在日時を再設定する必要があります。

【コマンド】 **S** *△ ddddtttt{cr}*

【動作】 現在日時をDWORD形式で設定する。

【引数】 *ddddtttt* ddddは、年月日を16ビット長で、ttttは時分秒を16ビット長でそれぞれ表します。

【解説】 内蔵のRTCに日時を設定します。設定した日時はファイル作成時にタイムスタンプとして使用されます。
Sコマンドで現在日時を送信後、"T△B"コマンドで計測を開始させます。
基板上のQ1に32.768KHzの水晶発振子を取り付けていれば、RTCが動作します。Vbatピンにバックアップ用のバッテリーを接続すれば、電源が切断されても時計計測はバックアップされます。

引数のddddttttは、DWORD型(32ビット)で現在日と現在時間を設定します。ビットの割り当ては下記の通りになっています。
下表の例では、2009年4月1日　18時30分00秒　と設定する場合についての値が書かれています。

ビット	内容	詳　細	例
31～25	年	現在の年から1980を引いた差	2009-1980=29 →2進数で 0011101 （7ビット長）
24～21	月	1月～12月	4月 →2進数で 0100 （4ビット長）
20～16	日	1日～31日	1日 →2進数で 00001 （5ビット長）
15～11	時	0時～23時	18時 →2進数で 10010 （5ビット長）
10～5	分	0分～59分	30分 →2進数で 011110 （6ビット長）
4～0	秒	秒を2で割った数　0～30	00秒 →2進数で 00000 （5ビット長）

例で2進数に換算した値を並べると、32ビットになるので、"00111010100000011001001111000000"となります。
これを16進数に換算すると、"3A8193C0"となりますので、"S 3A8193C0{cr}"として送信し現在日時を設定します。
なお、各進数への変換には関数電卓が便利です。Windowsにも標準でインストールされている"電卓"に関数機能がありますので、こちらを使用すると便利です。(※Windowsの電卓を使用する場合には、メニューバーの"表示"→"関数電卓"をクリックしてください)

この"S"コマンドで設定した日時を、MSC-MOD60が受信した時点から、設定日時からの時計計測が開始されます。
"S"コマンド送信の前に必ず"T△B"コマンドを送信してRTC機能を動作させておいてください。

※FATでは32ビットの時間管理が標準となっていますので、MSC-MOD60もこの方法に従っています。

【コマンド】 **G** △ **F** *{cr}*

【動作】 現在日時をRTCから取得する

【解説】 内蔵のRTCから現在日時を取得します。

【戻り値】 *!00* 最初にACKが返ります
mm/dd/yyyy – hh:mm:ss 月/日/年△-△時:分:秒 の形式で現在日時が返ります
!00 最後にACKが返ります

【コマンド】 **B** △*vvv{cr}*

【動作】 UARTの通信速度を設定

【引数】 *vvv* UARTの通信速度を下記の表を参考に指定します。

【解説】 UART通信の通信速度を切り替えます。
MSC-MOD60は工場出荷時は9600bpsに設定されています。下記の表に従い値を送信すると、通信速度が変わります。
なお、MSC-MOD60には、フルパワーモードと、省電力モードの2つの動作モードがあります。現在の動作モードによって値が変わりますので、現在の動作速度を確認後コマンドを実行してください。動作モードはZコマンドで変更できます。

通信速度(bps)	フルパワーモード時	省電力モード時
9600	DCEF	1FAB
19200	6EEF	0C7C
38400	37EF	067C
57600	43F2	08E5
115200	1EF4	04E5
230400	0FF4	×
460800	05A9	×
921600	028B	×

Bコマンドを送信すると、!00のACKが2つ返ります。最初に返る!00と、それに続くCRは変更前の通信速度で返ります。
2つ目の!00とそれに続くCRは、最初のACK完了後から約200ミリ秒後に新しく設定した通信速度で返ります。よって、新しい通信速度のACKを受信するためには、最初のACKを最初の通信速度で受信した後、200ミリ秒以内に新しく設定した通信速度で受信できるように設計します。なお、UARTで通信している場合には、データは取りこぼしても問題ありませんので、ACKは取りこぼしてしまっても問題はありません。通信速度が正しく設定できているか確認する場合には、新しい通信速度でCRを送信して、ACK(!00)が受信できれば速度が変更されていることが確認できます。

【使用例】 フルパワーモード時に、通信速度は115200bpsに変更する場合
B 1EF4*{cr}*
!00 設定前の通信速度で返ります。
!00 上の設定前のACKが送信されてから約200ミリ秒後に新しい通信速度で返ります。

【補足】 本設定は、電源を切断又はハードウエアリセットをすると初期値の9600bpsに戻ります。通信速度設定を記憶することはできません。
パワーモードを変更する場合にはZコマンドを使用します。パワーモードの変更はせずに通信速度だけの変更をする場合にはBコマンドを使用します。

【コマンド】 **I** *{cr}*

【動作】 装着されたSDカード又はMMCのデバイスのファイルシステムを初期化する

【解説】 装着されたメモリーカードのFATを初期化して、使用できるようにします。MSC-MOD60では、メモリーカードへのアクセスをはじめる前にIコマンドでファイルシステムの初期化をする必要があります。メモリーカードを入れ替えた場合にも行う必要があります。
Iコマンドを受信すると、MSC-MOD60はすべてのメモリーカードに対する設定が初期化されます。

【戻り値】 !00 初期化が正しく完了すると、ACK記号(!00)が返ります。
各種コマンドを続いて送信する際には、必ず!00が返ったことを確認してからコマンドを送信してください。
!00以外の数値が返る場合には、何らかのエラーが発生しています。エラーが発生した場合には、本書の"エラーコードについて"の項をご覧下さい。
!00は、メモリーカードの初期化が完了したことを意味します。

!05 SDカードが正しく挿入されていない場合や、不正なカードが挿入されている場合に戻ります。
カードの挿入状態を確認してください。

【補　足】 ■MSC-MOD60の電源が入っている状態でSDカードを脱着した場合
MSC-MOD60に電源が入った状態でSDカードを外した場合には、クローズ処理をしていないファイルハンドルの内容はすべて破棄されます。クローズ処理を行った後で外した場合には、ファイルへの編集内容や新規作成したファイルはSDカードに記録されています。
新たにSDカードをMSC-MOD60に装着した場合には、再度Iコマンドを送信してファイルシステムの初期化を行ってください。

■!BDなどのエラーが返り初期化ができない場合
Iコマンド送信後、!BDなどのエラーメッセージが返り、初期化が正しく完了できない場合には、Iコマンドを再度送信して、!00が返るまで送信を繰り返してください。送信は最大10回程度まで繰り返し、それでも初期化できない場合にはSDカードの不良やファイルシステムの破損、その他SDカードとの相性などが原因のことがあります。

【コマンド】 **K** *{cr}*

【動作】 SDカードの全容量と、空き容量のセクターサイズを取得する

【解説】 MSC-MOD60に挿入されているSDカードの全容量と、空き容量のセクター単位のサイズを16進数のDWORD型で返します。
単位はセクターです。1セクターは512バイトとして換算しますので、戻った値に512を乗算した値がバイト単位での値となります。
※サイズの大きなメディアの場合、取得までに時間がかかる場合があります。(2GBのメモリーの際の実測では約30秒程度です)
必ず戻り値が戻ってから次のコマンドを送信してください。

【戻り値】 下記の書式で結果が戻ります。
!00
$*ssssssss* $*ffffffff* *s* は全容量のセクターサイズ、*f* は空き容量のセクターサイズです
!00

例えば256MBのUSBメディアの場合、*s* は00079220となります。これは16進数ですので10進数に変換すると496160セクターですので、512バイト/1セクターを乗算すると、254033920バイトとなります。

【コマンド】 **@** *{cr}*

【動作】 ファイルリスト及びフォルダリストの初期化とリストの作成

【解説】 カレントディレクトリにあるファイルとフォルダのリストを一度初期化した後、新規にリストを作成します。Lコマンドでカレントディレクトリの内容を取得する場合にこのリストを使用します。そのため、@コマンドはLコマンド実行前に実行する必要があります。
※このコマンドだけではリストを見ることはできません。Lコマンドと組み合わせてご使用下さい。

【戻り値】 初期化が完了すると、!00のACKが返ります。

【コマンド】 **L △ A**{cr}

【動作】 カレントディレクトリに存在するフォルダ名及びファイル名を表示する （ASCIIコードで出力）

【解説】 カレントディレクトリにあるフォルダ及びファイル名を1つずつ表示します。表示は1つずつですので、複数のファイルやフォルダがある場合には、Lコマンドを複数回実行します。Lコマンド実行前には必ず@コマンドでリストを初期化しておきます。
カレントディレクトリ内のすべてのファイルやフォルダをを参照し終えた場合には !4D が返ります。

【戻り値】 *!00* ACKが返ります
$aa△$ssssssss△$nnnn 詳細は下記をご覧下さい
Name bytes!00 ファイル名又はフォルダ名です。名前の文字列はCRで終端されておらず、!00が付加されます

aa は1バイトサイズで属性データです、属性一覧に示す属性が返ります。値は16進数表記です。
ssssssss は4バイトサイズでファイルサイズをバイト単位で16進数表記で返します。フォルダの場合には00000000となります。
nnnn は2バイトサイズでファイル名のサイズを16進数表記で返します。（拡張子を含む）

属性は下記の通りです。

ビット	5	4	3	2	1	0
属性	アーカイブ	フォルダ	ボリュームID	システム	隠しファイル	読み取り専用

属性一覧（該当するビットが1になります）

※例えば、読み取り専用のフォルダの場合には、"010001"となりますので、戻り値の値は16進値として "11" と表示されます。
※MSC-MOD60でファイルやフォルダを作成する場合、それらに属性の設定はできません。

カレントディレクトリが、ルートディレクトリよりも下の階層にある場合には下記のようになります。

!00 ACKです。
$10 $00000000 $0001 属性は10(10000)となり、フォルダを示します。名前サイズは1バイトとなります。
.!00 .(ピリオド=0x2E)に続き!00が返ります。 ※!00の後ろにはCRが付加されます

ここで再度L Aを送信すると下記のような文字列が返ります。

!00 ACKです。
$10 $00000000 $0002 属性は10(10000)となり、フォルダを示します。名前サイズは2バイトとなります。
..!00 .(ピリオド)が2つと!00が返ります。 ※!00の後ろにはCRが付加されます

再度L Aを送信すると、そのディレクトリにあるファイル名やフォルダ名が返ります。
※ディレクトリの階層の深さに関わらずルートディレクトリよりも下にある場合には、常にこの内容になります。

【使用例】 カレントディレクトリに、アーカイブ属性の、約5.6MBの"music.mp3"というファイルがある場合

!00 ACKが返ります。
$20 $0055657A $0009 属性はアーカイブ、ファイルサイズは55657Aバイト=5596538バイト、ファイル名は9バイト
music.mp3!00 ファイル名に続いて!00が戻ります。

【コマンド】 **A** △ *foldername {cr}*

【動作】 カレントディレクトリを変更(移動)する

【引数】 *foldername* 移動したいフォルダ名を文字列で指定します。大文字小文字は区別されません。
なお移動できるのは1回のAコマンドで1階層です。複数の階層を移動する場合には、1階層ずつAコマンドで移動します。
一つ下の階層に移動するには、".." (ピリオド2つ 0x2E 0x2E)を送信します。

【解説】 カレントディレクトリとは、現在各種作業・操作ができるディレクトリのことです。
Aコマンドでは、指定した1階層上のフォルダにカレントディレクトリを移動させることができます。
下の階層へ移動したい場合には、 ".." (ピリオド2つ)で移動するとができます。
なお、現在のカレントディレクトリにあるファイル及びフォルダは@コマンドとLコマンドで確認することができます。

【戻り値】 正しく指定したフォルダに移動できると、!00のACKが返ります。
指定したフォルダが存在しない場合には、!41 が返ります。

【使用例】 SDカードのルートディレクトリ(一番上のディレクトリ)を root¥ として、
"root¥LOG¥RUN¥061210¥LAPTIME.LOG" というファイルを操作したい場合・・・ ※各行にはCRが付加されているとします。

A LOG
!00
A RUN
!00
A 061210
!00 ←この!00が送信された時点で、カレントディレクトリは、061210フォルダに移動しています

【補足】 ファイルの操作や、新しいファイルを新規作成したい場合には、各種操作のコマンドを実行する前に、目的のカレントディレクトリに移動しておく必要があります。特に新規にファイルを作成する場合には、必ずOコマンドを実行する前に、ファイルを作成したいフォルダに移動しておきます。なお、MSC-MOD60では大文字、小文字は区別されません。

【コマンド】 **M** △ *foldername{cr}*

【動作】 フォルダを新規に作成する

【引数】 *foldername* 新規に作成したいフォルダの名前を半角英数字で指定します。日本語などの2バイト文字は使用できません。

【解説】 カレントディレクトリに任意の文字列のフォルダを作成します。
フォルダを作成したいディレクトリにあらがしめAコマンドで移動しておき、Mコマンドでフォルダを作成します。

【戻り値】 フォルダが作られると!00のACKが返ります。

【補足】 フォルダ作成には1秒～3秒程度の時間がかかる場合があります。必ず、ACKが返ったことを確認してから、次の処理を行うよう設計してください。

【コマンド】 **O** △ *fh mode>filename{cr}*

【動作】　指定したファイルハンドルに、モードを指定してファイルをオープンする

【引数】　*fh*　0～3の数値　現在の空きファイルハンドルを指定します、ここで指定したファイルハンドルにファイルが展開されます

mode　"R"　"W"　"A" のいずれか1つのオープンモードを指定します

・・・"R" 読み込みモード

ファイルを読み込み専用モードで開きます。SDカード内にある既存のファイルを開きます。
開いたファイルからデータを読むことはできますが、データを書き込むことはできません。

・・・"W" 新規書き込みモード(ファイル新規作成)

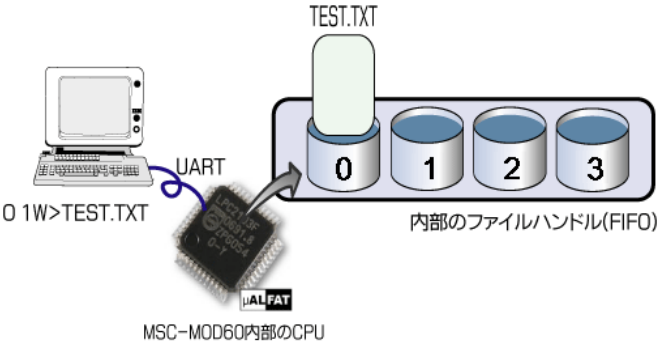
ファイルを新規に作成し、そのファイルに新たに書き込むモードです。
既存のファイルに書き込むのではなく新しくファイルを作成するモードです。

・・・"A" 追加書き込みモード

SDカード内にある既存のファイルへデータを追記するための上書きモードで開きます。
SDカードにある既存のファイルを開き、データを追加書き込みします。指定したファイルがない場合には、新規書き込みモードになります。

filename　開くファイルをファイル名で指定します。ディレクトリの指定はできませんので、ファイルが存在するディレクトリ又は、新規にファイルを作成するディレクトリを、先にAコマンドにてカレントディレクトリにしておきます。
カレントディレクトリに存在しないファイル名を指定した場合には、エラーとして !41 が戻り値として戻ります。

【解説】　MSC-MOD60ではファイルを扱う際には必ずファイルをファイルハンドルと呼ばれる専用のメモリ空間に開く必要があります。



Oコマンドは指定したファイルをファイルハンドルに開きます。ファイルハンドルはMSC-MOD60内に0～3まで4つあり、任意のファイルハンドルにファイルを開くことができます。
新規にファイルを作成する場合には、先にファイルハンドルに対してファイルを作成しファイルハンドルを閉じる際(Cコマンド)にファイルハンドルの内容をSDカードに書き込みます。

ファイルをファイルハンドルに開く際には、そのファイルをどんな目的で開くのかを指定します。これをオープンモードと呼びます。オープンモードには下記の3つがあります。
・読み込みモード
・新規ファイル作成書き込みモード
・既存のファイルへの上書きモード
ファイルを開く際には、どのモードでファイルを開くのかを決め、Oコマンド発行時に引数として指定します。
読み込みモードで開いたファイルには、データの追記はできません。また、新規ファイル作成書き込みモード及び既存ファイルへの上書きモードで開いたファイルからはデータを読み取ることはできません。モードを変更するには、一度ファイルハンドルを閉じてから(Cコマンド)改めてモードを設定してファイルを開きます。

Oコマンドで対象になるファイルは、常にカレントディレクトリに存在するファイルです。よって、Oコマンドでファイルを開く場合には、必ず目的とするファイルが存在する、又はファイルを作成したいディレクトリにカレントディレクトリを移動させておく必要があります。カレントディレクトリの移動は、Aコマンドで行うことができます。

【使用例】 ①カレントディレクトリにある TEST.TXT を読み込み専用モードでファイルハンドル1に開く場合

O 1R>TEST.TXT*{cr}*
!00

②カレントディレクトリに NEW.LOG ファイルを新規に作成し、そのファイルをファイルハンドル2に開く場合

O 2W>NEW.LOG*{cr}*
!00

③カレントディレクトリにある、LAP.LOGファイルに修正を加えるため、追加書き込みモードでファイルハンドル1に開く場合

O 1A>LAP.LOG*{cr}*
!00

【戻り値】 !00 正しくファイルハンドルにファイルが展開されたことを通知します。ACKです。
!4E データの存在しない0バイトのファイルを開こうとしました、データの存在するファイルを指定してください
!4F 指定したファイルハンドルは既に使用されています、別のファイルハンドを指定してください
!41 指定したファイルは存在しません
!31 使用できない文字列を含んだファイル名です、このファイルは開くこと及び作成することはできません
!33 拡張子が不正です、拡張子は3文字にしてください

※その他の戻り値については、本書の「エラーコマンド一覧」をご覧下さい

Oコマンドでファイルをファイルハンドルに開く場合には必ず戻り値を確認する必要があります。次の操作を行う場合には、必ず!00のACKが返ったことを確認してください。!00以外の戻り値が返る場合には、但しファイルが作れなかったり、ファイルからデータを読み出すことができません。ACKの確認を必ず行うようシステムの設計をお願い致します。

【補足】 MSC-MOD60では、ファイル名及びフォルダ名の大文字、小文字は区別されません。

Wモードでファイルハンドルにファイルを作り、データを書き込んだ場合、クローズコマンド(Cコマンド)を実行するまでは実際にはメディア上にファイルは作成されません。Cコマンドでファイルハンドルを閉じた時に初めてメディア上にデータを読み出すことのできるファイルが作られます。よって、Cコマンドを実行しないで電源を切断したり、SDカードを抜いたりすると、ファイルハンドルの内容は消えメディア上にもファイルが作られません。

内蔵のRTCを有効にしている場合には、Wモード及びAモードで作成、編集したファイルにはタイムスタンプが記録されます。

Oコマンドでファイルハンドルにファイルを開いた直後は、Wモード並びにRモードの時は、カレントカーソル位置は必ず0番地となります。すなわちファイルの先頭位置にカーソルが置かれます。Aモードで開くと、カレントカーソル位置は必ず最終番地の1つ後ろとなります。カレントカーソル位置を移動したい場合には、Oコマンドでファイルハンドルにファイルを開いた後、Pコマンドを使用してカーソル位置を移動させることができます。

アペンドモード(A)でオープンしようとした際に、指定したファイルが存在しない場合には追記モードにならず新規にファイルを作成します。(※Wモードと同等となります。)

【コマンド】 **C** △ *fh{cr}*

【動作】 指定したファイルハンドルをクローズして、Wモード及びAモードでオープンしているファイルをSDカードに書き込む

【引数】 *fh* 0～3 クローズしたいファイルハンドルを指定します

【解説】 ファイルハンドルを閉じます。Rモードで開いている場合にはそのままファイルハンドルを閉じます。
Wモード及びAモードで開いている場合には、データをSDカード上に書き込みファイルの実体を作成し、ファイルハンドルを閉じます。
ファイルは、クローズコマンドでファイルハンドを閉じたときに初めてその実体がファイルとして保存されます。よって、クローズコマンドでファイルハンドルを閉じる前にSDカードを本体から外したり、電源を切断したり、リセットをしてしまうと、ファイルハンドルに展開されているファイルはすべて破棄されてしまいますのでご注意ください。

【戻り値】 ファイルハンドルが正常に閉じると !00 が返ります。!00が返る前に電源の切断やSDカードの取り外しを行うと、ファイルは作成されません。必ず!00が返ったことを確認するようなシステムを設計してください。

【補足】 クローズコマンドを実行しないで、SDカードを本体から外したり、電源を切断したり、リセットをしてしまうと、ファイルハンドルに展開されているファイルはすべて破棄されてしまいます。電源切断が予期できないようなアプリケーションを作成される場合には、頻回にクローズコマンドを実行して、ファイルを作成させることをお奨めします。
※一度作成したファイルに対して上書きする場合には、アペンドモードでファイルをオープンします。

【コマンド】 **R** △ *fhM>ssssssss{cr}*

【動作】 指定したファイルハンドルのファイルの内容を指定したバイト数読み込む

【引数】 *fh* 0～3 読みたいファイルのあるファイルハンドルを指定します

M 超過文字を指定します。超過文字とは、読み取りを指定したバイト数よりも実際のデータのサイズが小さい時に、その差分を表す文字のことです。Mで指定した半角英数字1文字で表します

ssssssss データを読み込むサイズを16進数のバイト単位で指定します
"FFFFFFFF"まで指定できます。すなわち4294967295バイトまで1回のコマンドでデータを読み込めます

【戻り値】 戻り値は、読み取ったデータの内容と、ACK及び読み取りができたデータサイズです。
書式は下記のようになります。

!00	ACKが戻ります。
ddddd$ssssssss	*d* は読み込んだデータ、*s* は読み込んだデータのサイズの16進値です。
!00	最後にACKが返ります。

読み込んだデータの最後には必ず$マーク(0x24)が挿入されますので、データの最後を検出するのに使用できます。
なお、読み取り指定したサイズが、ファイルハンドルにあるデータのサイズより大きい場合には、最終データまで読み取った後、引数*M* で指定した超過文字が超過分付加されます。

その他のエラーコマンドについては、本書の「エラーコマンド一覧」をご覧下さい。

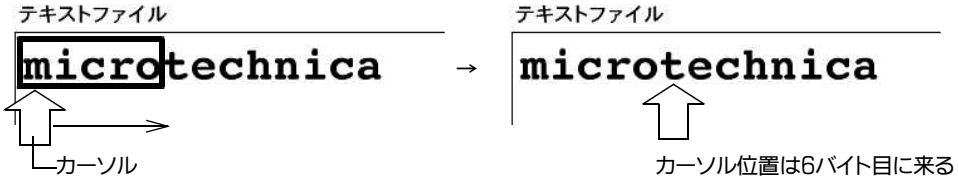
【解説】 Rコマンドは指定したファイルハンドルにRモードで開かれているファイルから指定したサイズ分データを読み取るコマンドです。
Rコマンドを使用する前に、必ず"0"コマンドにて、ファイルハンドルにファイルを読み込みモード(Rモード)で開いておく必要があります。
※Rモード以外のモード(WモードやAモード)で開いたファイルからはデータの読み込みはできません。

引数の*M*は超過文字を指定します。例えば5バイトしかないファイルに対して、7バイトの読み取りを指定した場合2バイト分が超過しています。そこでMSC-MOD60は、超過している2バイト分を*M*で指定した文字で返します。

Rコマンドで一度ファイルを読み込むと、カレントカーソル位置(処理を開始する位置のこと)が1回前に読み込みが終わった位置の次の番地に移動しています。例えば、ファイルハンドル1に文字列 "microtechnica" が書かれた13バイトのテキストファイルがあるとして、次の例をご覧ください。

R 1Z>5
!00　　ACKが返ります。
micro$00000005　　データ5バイトと、読み取ったデータサイズが16進数表記で戻ります。
!00　　最後にACKが戻ります。

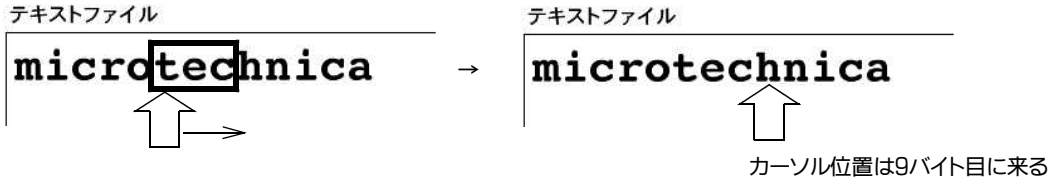
Rコマンド初回の実行ですのでカレントカーソルは先頭にあり、そこから5バイト分読み取りますので"micro"の5文字(5バイト)が返ります。(下記例では便宜上テキストファイルで文字列としてデータを表しています。)



続いて再度同じファイルハンドル1のファイルから3バイトのデータを読み取ります。

R 1Z>3
!00
tec$00000003
!00

すでにカレントカーソルは前回のRコマンドの実行によって6バイト目に来ていたため、次にRコマンドを実行すると、前回読み取りを終了した次の位置からデータが読み取られます。カレントカーソルの位置は、Pコマンドにて任意の位置に設定できます。



続いて同じファイルハンドル1のファイルから10バイトのデータを読み取ります。

R 1Z>A
!00
hnicaZZZZZ$00000005　　9バイト目から残り5バイトのデータが返り、指定超過分の5バイトを示す超過文字Zが5つ返ります
!00

データを読み終わってもカーソルは先頭には戻らず最終位置で止まります。先頭に戻したい時はPコマンドを使用して先頭にカーソル位置を戻します。

※改行を含むデータを読み出す場合
読み出すファイル内に改行コードが入っている場合でもデータはそのまま改行コードをデータとして読み出します。
Windowsで作られてたテキストファイルの場合には、改行はCRとLFの2バイトです。

【使用例】 ※ファイルハンドル1のファイルに "12345678901234567890" というデータがある場合・・・

①ファイルハンドル1にあるファイルから、先頭番地から10バイト分のデータを読む

R 1Z>A*{cr}*
!00
1234567890$0000000A
!00

②4バイト目から7バイトデータを読み込む

P 1>4*{cr}*　　Pコマンドはカレントカーソルの位置を任意の位置に指定できるコマンドです
!00
!00
5678901$00000007
!00

③先頭番地から25バイト分データを読み込む

R 1Z>19
!00
12345678901234567890ZZZZZ$00000014　　*Z*は超過文字として超過したデータ分を表示しています
!00

【コマンド】 **W** △ *fh>ssssssss{cr}*

【動作】 指定したファイルハンドルにWモード又はAモードで開かれたファイルに、指定したサイズ分のデータを書き込む

【引数】 *fh* 0～3 データを書き込みたいファイルの開かれているファイルハンドルを指定します

ssssssss 書き込むデータサイズを16進数表記のバイト単位で指定します
"FFFFFFFF"まで指定できます。すなわち4294967295バイトまで1回のコマンドでデータを書き込むことができます

【解説】 ファイルハンドルにOコマンドのWモード又はAモードで開かれているファイルに対して、データを書き込みます。
Rモードで開かれたファイルに対しては書き込みはできません。
MSC-MOD60はWコマンドを受信後、直ちに!00を返し、データの受信待機状態となります。この!00がMSC-MOD60から送信されたことを確認してから、*ssssssss*で指定したサイズのデータを送信します。!00以外のコードが返った場合には、正しく書き込みができませんので、必ずその原因をエラーコードから推測して処置を講じて下さい。

MSC-MOD60は、Wコマンドの*ssssssss*で指定したバイト数のデータを受信するまで永久的にデータの受信を待機します。
よってWコマンドを実行する場合には、書き込むデータのサイズをあらかじめよく考慮した上で書き込むデータサイズを指定する必要があります。指定したデータサイズに達しない場合、いつまでもMSC-MOD60はデータの到達を待ちます。(タイムアウト時間の設定はできません。)クローズコマンド(C)を送信しても、データとして受け付けてしまいますので、ファイルハンドルを閉じることはできません。

Oコマンドにて Wモード(新規ファイル作成モード)でファイルをファイルハンドルに開いている場合には、必ず1バイト目(ファイルの先頭)から書き込みが開始されます。Aモード(上書きモード)でファイルをファイルハンドルに開いている場合には、必ず最終番地の次の番地から書き込みが開始されます。データを書き込む番地(位置)を変更したい場合には、Pコマンドでカレントカーソルの位置を変更することができます。

Pコマンドを使用することで任意の番地(位置)にデータを書き込むことができますが、既にデータが存在する位置にカーソルを移動してデータの書き込みを行うと、データは上書きされます。(データを挿入することはできません)下記に例を示します。

```
O 1W>TEST.TXT        ファイルハンドル1に新規にTEST.TXTというファイルをWモードで作成
!00                  ACK
W 1>5                5バイトのデータ書き込みを指定
!00                  ACK
12345                書き込むデータを送信　5バイト
$00000005            書き込み完了
!00                    〃
```

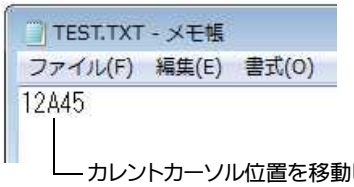
この時点で"C 1"でファイルハンドルを閉じてデータを見ると次のようになっています

閉じずに続けて操作をした場合・・・

(Windowsのメモ帳で開いた場合)

```
P 1>2                カレントカーソル位置を2バイト目に指定
!00                  ACK
W 1>1                ファイルハンドル1のファイルに1バイト(1文字)のデータを書き込み指定
!00                  ACK
A                    文字'A'をデータとして書き込み
$00000001            書き込み完了
!00                    〃
C 1                  ファイルハンドル1を閉じる
!00                  ACK
```

ここで作られたファイルを見ると次のようになっています。



カレントカーソル位置を移動したため、データが上書きされています。
※挿入ではなくデータが上書きされることにご注意下さい。

ファイルにデータがすべて書き終わった場合には、必ずCコマンドにてファイルハンドルをクローズします。クローズ処理を行うことで初めてファイルの実体が作成されます。(データを読み出すことのできるファイルとなります。)
Cコマンドによるクローズ処理をしないで電源を切断したりSDカードを外すと、データはすべて破棄されてしまいますのでご注意下さい。

【戻り値】 MSC-MOD60は、指定したサイズ分のデータを受信すると、 下記のような戻り値を返します。

$ssssssss
!00

$に続き、*ssssssss* にて実際に書き込んだデータサイズを16進数表記の文字列で表示します。
また書き込みが成功したことを示すため !00 のACK記号が1回返ります。
もし送信したデータの最後がキャリッジリターン(CR=0x0D)で終端されていた場合には、このCRに対応するACK(!00)がもう1つ返ります。テキストデータを送信している場合などは、ターミナルソフトなどの仕様で意図せずデータの最後にCRが付加されている場合があり、その場合には!00が2回返ります。

【使用例】 ファイルハンドル1にWモードで開かれているファイルに対し、文字列 microtechnica を書き込む場合

W 1>D*{cr}*	書き込むサイズは16進数で表現 (0x0Dバイト = 13バイト)
!00	Wコマンドを正しく受信
microtechnica	書き込むデータを送信(Wコマンドで指定したデータサイズ分を送信します)
$0000000D	13バイトの書き込み完了の戻り値
!00	書き込み正常終了
C 1*{cr}*	ファイルハンドル1を閉じる
!00	ファイルハンドル1のクローズ処理完了

※上記の例では、13バイトのデータを書き込んだ後クローズ処理をしていますが、これは一例であり、続けて再度Wコマンドを実行してデータを書き込むことができます。

※文字列 microtechnica がCRで終端された場合には、CRというデータ1バイト分に相当する!00が1回返ります。

【応用】 <u>※書き込むデータに改行を挿入する場合</u>
書き込むデータに改行を挿入したい場合には、改行コードをデータとして送信します。改行コードの挿入されたファイルを閲覧する場合、OSによって改行コードの取り扱いが異なります。下記に一覧を示します。

LF	UNIX系のシステム。Linux、Mac OS X、BeOS、Amiga、RISC OSなど
CR+LF	MS-DOS、Microsoft Windows
CR	Apple IIファミリ、Mac OS(バージョン9まで)

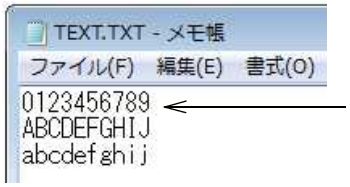
※LFは0x0A、CRは0x0Dです。

Windowsにて該当のファイルを読む場合、改行として認識させるためには、CRとLFの2バイトが挿入されてる必要があります。
CR(0x0D)しか、入っていないファイルをWindows標準のメモ帳で開くと改行が正しく反映せず、改行位置に■のような記号が入ります。
インターネットエクスプローラーや表計算ソフトのエクセルなどでは正しく改行が反映されますが、Windowsで扱うテキストデータを作る場合には改行には、CR+LFの2バイトを挿入することをお奨めします。

【応用例】 ファイルハンドル1に新規にファイル(TEXT.TXT)を作成して改行を含むデータを作成した場合の例

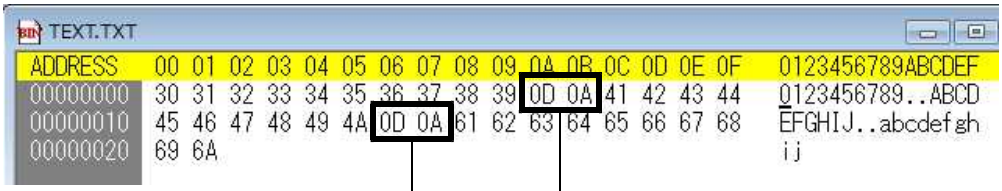
O 1W>TEST.TXT	
!00	
W 1>22	34バイトのファイルサイズを指定
!00	
0123456789(CR)(LF)ABCDEFGHIJ(CR)(LF)abcdefghij	CRとLFを含むデータを送信
$00000022	書き込み完了
!00	〃
C 1	
!00	ファイルの実体作成

ここまでて作成されたファイルをWindowsのメモ帳とバイナリエディタで開くと次のようになっています。



改行が入っていることが分かります。

メモ帳で開いた場合



バイナリエディタで開いた場合

改行コード0x0D(CR)と0x0A(LF)が挿入されています。

【コマンド】 **P** △ *fh>ssssssss{cr}*

【動作】 ファイルハンドルのカレントカーソルの位置を任意の位置に設定する

【引数】 *fh* 0～3 ファイルハンドルを指定します
ssssssss 移動したいカレントカーソルの位置を16進数表記の文字列で指定します

【解説】 カレントカーソルの位置は、引数*ssssssss*で指定します。表記は16進数表記です。カレントカーソルの位置を指定するとでRモードで開かれたファイルであればデータの読み込み開始位置を、Wモード及びAモードで開かれたファイルであればデータの書き込み位置を指定することができます。データの先頭に移動する場合には、0を指定します。

【戻り値】 *!00* 指定した位置に正しくカレントカーソルが移動できました
!53 カレントカーソルの位置指定が不正です、正しい値が設定されていません。指定した位置がデータサイズを超えています。
!4F ファイルハンドルの指定が不正です。

【使用例】 ファイルハンドル1のカレントカーソルを15バイト目に移動する場合

P 1>F*{cr}*
!00

【コマンド】 **D** △ *filename{cr}*

【動作】 指定したファイルを削除する

【引数】 *filename* カレントディレクトリにある削除したいファイルのファイル名を拡張子も含めて指定します

【解説】 カレントディレクトリにある任意のファイルを削除します。Dコマンドで削除できるファイルは、カレントディレクトリにあるファイルだけですので、別のディレクトリにある場合にはAコマンドでカレントディレクトリを移動します。
※フォルダの削除はEコマンドで行って下さい。Dコマンドはファイルの削除だけです。

正しく削除が完了すると、ACKの!00が返ります。

【コマンド】 **E** △ *foldername{cr}*

【動作】 指定したフォルダを削除する

【引数】 *foldername* カレントディレクトリにある削除したいフォルダ名を指定します

【解説】 カレントディレクトリにある任意のフォルダを削除します。削除できるフォルダは、カレントディレクトリにあるフォルダだけですので、別のディレクトリにある場合にはAコマンドでカレントディレクトリを移動します。
※ファイルの削除はDコマンドで行って下さい。Eコマンドはフォルダの削除だけです。

正しく削除が完了すると、ACKの!00が返ります。

【コマンド】 **Q △ CONFIRM △ FORMAT***{cr}*

【動作】 装着されたメモリーカードをクイックフォーマットする

【解説】 SDカードやMMCのクイックフォーマットを行います。MSC-MOD60ではローレベルフォーマットはできません。ファイルシステムのファイル管理情報の保存領域のみを消去します。
フォーマットすると、メモリーカード内のすべてのデータは削除されます。
フォーマットには時間がかなりかかり、その所要時間はメモリーカードサイズに依存します。

【戻り値】

Q CONFIRM FORMAT	クイックフォーマットを実行します
!00	最初のACKはクイックフォーマットの開始を通知するものです
!00	2回目のACKはフォーマットが完了したことを通知するものです

2回目のACKが返るまでの間がクイックフォーマットしている時間です。2回目のACKが返るまではメモリーカードの抜き差しをしないようご注意下さい。

【補足】 このクイックフォーマットはあくまでも、ファイルの管理を行う管理情報の保存領域のみのフォーマットを行うものですので、見かけ上このコマンド実行後はファイルやフォルダはなくなったように見えますが、データそのものはメモリーカード内に存在しています。
また、セクターサイズの変更もできません。
クイックフォーマットに要する時間はメモリーカードの容量や、メーカー等によってかなり変動します。実際に使用するメモリーカードを使用して実測されることをお奨めいたします。長い場合3分程度かかる場合もあります。

【コマンド】 **?** △ *filename{cr}*

【動作】 指定したファイル名のファイル又はフォルダ名のフォルダが、カレントディレクトリに存在するかどうかを確認する

【引数】 *filename* 調べたいファイル名又はフォルダ名を文字列で指定します。 ワイルドカードは使用できません。
完全一致したファイル又はフォルダだけ検出します。

【戻り値】

・指定した文字列のファイルがカレントディレクトリに存在していた場合

!00	ACKが返ります
$ssssssss△$AA△$ddddtttt	*ssssssss* 見つかったファイルのファイルサイズが16進数表記で返ります
!00	*AA* はファイルの属性が戻ります
	ddddtttt は該当ファイルの最終更新日時を32ビットのDWORD型で返します
	最後にACKが返ります。

└ 本書10ページをご覧下さい

・指定した文字列のファイルがカレントディレクトリに存在しなかった場合

!41	見つからなかった場合にはエラーとして!41が返ります。

【解説】 カレントディレクトリに指定したファイルやフォルダが存在しているかを検査するためのコマンドです。
ファイルが見つかった場合には、そのファイルのファイルサイズ・属性・最終更新日時(DWORD型)を返します。
見つからなかった場合には、!41を返します。

【コマンド】 ～{cr}　　※チルダです。ASCIIコードは0x7E

【動作】 ロングファイルネーム規則(LFN)のライセンスを取得済みの製品かどうかを判定します。
バージョン3.12以降のファームウエアであっても、旧製品からのアップデートの場合にはライセンスを取得していない場合があります。
その場合、ファームウエアを更新してもロングファイルネームは使用できません。
本コマンドはお使いのMSC-MOD60が、LFN規則ライセンスを取得しているかどうかについて確認します。

【戻り値】

!00	ACKが返ります
$aa	aaの部分が、00の場合にはLFNは使用できません。01の場合にはLFNが使用できます。
!00	ACKが返ります。

【補足】 ロングファイルネーム規則は、マイクロチップ社の所有するFATライセンスによるもので、使用に際してはライセンスを受けていることが必要です。本製品は、マイクロチップ社からライセンスを取得していますが、その確認を行うためのコマンドです。
旧製品で、ライセンスを取得する以前の製品の場合には、ロングファイルネーム規則は使用できません。ファームウエアのバージョンが最新版でもライセンスを取得していない場合、ロングファイル名を使用しようとすると、!32などのエラーが返ります。

【コマンド】 X △ S{cr}

【動作】 ファームウエアをアップデートするためのコマンドです。MSC-MOD60は将来のファームウエア更新のために、アップデートを行うコマンドを用意しています。新しいファームウエアが公開されると、当方のWEBページで告知致します。下記の手順でアップデートを行うことができます。MSC-MOD60を機器に組み込む場合には、将来アップデートができるように本コマンドが発行できるように設計して頂けますようお願いいたします。

【手順】 ①当方のWEBサイトから最新版のファームウエアをダウンロードします。
ファイル名は、"UALFATFW.GHI"となっています。このファイルをフォーマット後のSDカードにコピーしてください。

②MSC-MOD60に①のSDカードを装着します。

③シリアルコマンドで、"X△S{cr}"のコマンドを送信します。
※メモリーカード初期化のIコマンドは送信するは必要ありません。
※シリアル通信はUARTでもI2Cでも可能ですが、I2C通信を使用している場合には、アップデート処理中にMSC-MOD60から返る不定長のデータをすべて遅滞なく受信するように設計してください。

④アップデートは3秒～5秒程度で完了します。アップデートが完了すると、本体起動時又はリセット時に表示されるメッセージがシリアル通信経由で送られてきます。これでアップデートは完了です。

【補足】 現在のファームウエアのバージョンは、"V"コマンドで確認できます。
アップデート実行中は絶対に電源を切断したり、シリアルコマンドを送信したり、SDカードを抜いたりしないでください。アップデートに失敗してしまう場合があります。

<u>■アップデートに失敗した場合</u>
アップデートに失敗することはほとんどありませんが、もし失敗してしまった場合そのままではMSC-MOD60は動作しなくなります。
但し、MSC-MOD60にはファームウエアの領域の他にブートローダー領域があり、ブートローダーはそのまま保護されています。ブートローダーを使用することによって、"ブートローダーモード"と呼ばれる特殊なモードでファームウエアをダウンロードすることができます。
アップデートに失敗した場合には、"I"コマンドやCRなどを送ると、"!DE"が返ります。

①MSC-MOD60のモード1ピン・モード2ピンを次のようにします。
モード1ピン・・・1　　※1はHの状態(電源電圧+3.3Vと接続)、LはLの状態(GNDと接続)です。
モード2ピン・・・0

②ファームウエアを保存してあるSDカードをMSC-MOD60に装着します。

③シリアル通信で、"LOK{cr}"(エルオーケー、すべて大文字)を送信します。

④ブートローダーモードでアップデートが開始されます。アップデートが完了したら、①で設定したモードピンの設定を、シリアル通信モードの設定に直してください。(本書3ページを参照してください)

エラーコード一覧

MSC-MOD60は送信したコマンドに対してエラーが発生するとエラーを返します。の内容は下記の通りです。

※エラーコードはあくまでもエラーの目安であり、時に不正確な(下表に記載でない内容)エラーが返ることがあります。

※表中に*印のあるものは、メモリーデバイスのフォーマットが必要です。フォーマットはWindowsでクイックフォーマットではなく、通常のフォーマットを行って下さい

エラーコード	エラーの内容
0x01	読み込みセクターエラーです
0x02	書き込みセクターエラーです
0x03	消去セクターエラーです
0x04	メモリーカードが不正な値を返しました
0x05	メモリーカードの初期化でタイムアウト発生です
0x06	ブロックサイズの設定エラーです
0x07	メモリーカードへのコマンド送信でエラーです
0x10	ブートセクターエラーです　*
0x11	MBRでエラーです　*
0x12	ブートセクターエラーです　*
0x13	セクターサイズエラーです　*
0x14	ファイルシステムエラーです　*
0x15	サポートしていないファイルシステムです
0x16	サポートしていないファイルシステムです
0x21	クラスターにエラーがあります　*
0x22	クラスターにエラーがあります　*
0x23	クラスターにエラーがあります　*
0x24	クラスターにエラーがあります　*
0x25	メモリーメディアがいっぱいです
0x31	ファイル名に書式上の誤りがあります
0x32	×　削除
0x33	×　削除
0x34	ファイル名に書式上の誤りがあります
0x35	メモリーメディアがいっぱいです
0x40	ファイル又はフォルダ名が既に存在しています
0x41	ファイル又はフォルダ名が見つかりません
0x42	フォルダーに不正な情報がありますA　*
0x43	フォルダーに不正な情報がありますB　*
0x44	フォルダーに不正な情報がありますC　*
0x45	FAT16ではルートディレクトリに512個のエントリーのみ存在できます
0x46	ファイルのオープンに失敗しました
0x47	読み込みモードのため、書き込みはできません
0x49	Pコマンドで指定できる値は、最大ファイルサイズまでです。ファイルサイズ以上が指定されました
0x4A	フォルダが空ではありません、フォルダ消去前にはフォルダ内が空である必要があります
0x4B	この名前はフォルダではありません
0x4C	Rモードで開くことを要求します
0x4D	ファイル・フォルダリストが最後まで達しました
0x4E	ファイルパラメーターが不正です　*
0x4F	すでに使用されているファイルハンドルです
0x50	ファイルサイズが0です
0x51	ファイルモードが不正です
0x52	ファイルが破損しています
0x53	ポインターがファイルサイズを超えています
0x61	不正なコマンドです、存在しないコマンドです
0x62	不正なコマンドです、予期しないエラーです
0x63	不正な名前です
0x64	不正な数字です
0x65	Wコマンドで処理が失敗しました
0x67	メモリーメディアのオープンに失敗しました
0x68	パラメーターが正しくありません
0x69	チェックサムエラーです
0x71	USBパイプが不足しています　※1
0x72	ハンドルはすでに使用されています　※1
0x73	USBデバイスが不正なディスクリプタを返しました　※1
0x74	転送機能が制御できません
0x75	エンドポイントの設定においてエラーが発生しました　※1
0x76	USBレスポンスがタイムアウトしました　※1
0x77	コントロール転送が要求されました
0x78	USBデバイスがNACKを返しました　※1
0x79	USBハンドルが不正です　※1
0x7A	USBディスクリプターが不正です　※1
0x7B	ディスクリプターが見つかりません　※1
0x7C	ハブデバイスが見つかりません　※1
0x7D	HCDが接続されていません　※1
0x81	USBストレージデバイスが処理に失敗しました ※1
0x82	USBストレージデバイスが処理に失敗しました ※1
0x83	USBストレージデバイスが処理に失敗しました ※1
0x85	USBストレージデバイスが処理に失敗しました ※1
0x86	USBストレージデバイスが処理に失敗しました ※1
0x90	MAX3421の初期化に失敗しました　※1
0x91	HCDでエラーが発生しました　※1
0xA0	ストレージデバイスの準備ができていません　※1
0xA1	サポートされていないプロトコルです　※1
0xA2	サポートされていないサブクラスです　※1
0xA3	予期しないエラーです　※1
0xA4	USBストレージデバイスが不正な応答をしました※1
0xB1	HCDがビジーです　※1
0xB2	HCDの不正なリクエストです　※1
0xB3	HCDで定義されていないエラーです　※1
0xB4	HCDからNAKが返りました　※1
0xB5	HCDがハングアップしました　※1
0xB6	HCDでエラーが発生しました　※1
0xB7	HCDでエラーが発生しました　※1
0xB8	HCDでエラーが発生しました　※1
0xBA	HCDでエラーが発生しました　※1
0xBB	HCDでエラーが発生しました　※1
0xBC	HCDでエラーが発生しました　※1
0xBD	HCDでエラーが発生しました　※1
0xBE	HCDでエラーが発生しました　※1
0xBF	HCDでエラーが発生しました　※1
0xC0	名前が長すぎます
0xC1	先頭文字が不正です
0xC2	無効な文字列が含まれています
0xC5	名前の最長に到達しました
0xD0	不正なアドレスレンジです
0xD1	フラッシュメモリーはブランクではありません
0xD2	ベリファイエラーです
0xD3	内部エラーです

0xD4	チェックサムエラーです
0xD5	イレース処理でエラーです
0xD6	アクティブシークエンスでエラーです
0xD7	ファームウエアが無効です
0xD8	不正なコマンドです
0xDA	空のファイルです
0xDB	ファイルが見つかりません
0xE5	デバイスがみつかりません
0xF0	予期しない値です
0xFD	定義されていないコマンドです

※1のエラーはUSBH-ACS20(USBフラッシュメモリー用ボード)のエラーですので、本MSC-MOD60には直接関係ありません。

■エラー発生時の対応方法

MSC-MOD60では、ほとんどのコマンドで正しく処理が完了したり、コマンドを正しく受信すると、ACKとして!00が返るような仕組みになっています。詳しくは各コマンドの詳細に記述されていますが、多くの場合、MSC-MOD60に接続するシリアル通信側の機器は、この!00を待ってから次の操作を行う必要があり、!00のACKを待たないで、又はACKの受信をしないで次の操作を行うことは、様々な問題に繋がります。

MSC-MOD60が!00以外のエラーコードを返す場合には何らかのトラブルが発生していることを示しており、エラーコードが返ってくる以上正常な動作を期待できません。よって、MSC-MOD60と接続する機器はこのエラーコードを受信したら、処理を一度停止し原因を解決しなくてはなりません。原因は特定が容易なものから特定ができない難しいものまで含まれています。多くの場合には次のような方法でエラーに対処することが望ましいと考えられます。

①エラーコード表からエラーを特定する

エラーコード表を参照して原因の特定を試みます。原因が分かれば、それに対応した方法でエラーを回避するようにします。

②エラーが解決しない場合

エラーは各機器がすべて正常に動作していても、タイミングのずれや、信号に物理的な変動などの不確定要因が生じていたり、外部からのノイズなど、様々な要因で発生することがあります。それらのエラーであった場合には、同じルーチンを繰り返し実行することで、正常に動作することがあります。エラーが発生している場合で、エラーの原因が特定できないような場合には、再度同じコマンドを発行することで問題が回避できるかもしれません。少なくとも10回程度は同じコマンドを発行してみてACKが返ってくることがあるかどうかについて確認をしてみてください。

③本体をハードウエアリセットする

エラーが発生し続けている場合には、一度MSC-MOD60本体のハードウエアリセットをすると問題が回避できるかもしれません。リセットピンは内部でプルアップされていますので、定常状態ではオープン、又はHレベルにしておきますが、リセットをかける場合には、最小30ミリ秒以上のLレベルのパルスを印加してリセットを行い、再度試行してみてください。

④電源を再起動する

ハードウエアリセットをかけても、本体のCPUはリセットされますが、取り付けたメモリーデバイスはリセットされていません。電源を再起動することでメモリーデバイスもリセットされますので、③試行後でもエラーが続く場合には一度本体を再起動することが有用である場合があります。

⑤メモリーデバイスのフォーマットをしてみる

メモリーデバイスへの書き込みや読み込みは、FATというファイルシステム仕様に基づいて行われています。ファイルシステムに問題が生じていたり、正しくフォーマットができていないメモリーデバイスを使うと、エラーが発生したり予期しない問題が発生することになります。一度Windowsでメモリーデバイスをフォーマットし直してお試し下さい。なお、フォーマットを行う場合には次の設定としてください。

・フォーマットは通常フォーマット(クイックフォーマットは使わない)
・SDフォーマットをしてみる

⑥相性問題を疑ってみる

パソコンなどで使用されるメモリーデバイスは汎用的な製品であり、ある規格に即して設計、製造されています。よって一般的にはその規格を満たす機器同士であれば正常に使えるはずです。
しかし、個別の機器自体は正常に動作しており、規格に即した仕様になっているにも関わらず、組み合わせて使用すると正しく動作しなかったり動作が不安定だったり・・という問題が発生することがあります。これを一般的には相性問題と呼んでいますが、規格がオープンで広く普及した規格では相性問題がより起こりやすいと言えます。
相性問題が発生する理由は、定められた規格の仕様に一定の余裕が持たせてあるためです。例えば時間的なタイミングにしても、最小〇〇μ秒～最大〇〇μ秒まで・・・というように仕様に幅があるものです。
そのため仕様範囲内であっても、タイミングのずれが重なると、許容範囲を逸脱してしまったり、動作に問題が生じたりすることがあります。
また、高速で動作する機器の場合には、分布定数回路が形成されて、波形が歪んだり、波形の立ち上がりや立ち下がりになまりが生じて、その結果、問題が発生するということもあります。
相性問題は、汎用的な規格を用いる場合にはなかなか避けられない問題で、その問題が発生した場合には解決の方法が少ないのも事実です。そういった場合には機器を交換するしかありません。
もし、動作がどうしてもうまくいかなかったり、不規則な問題が生じる場合には相性問題を疑って頂き、使用するメモリーデバイスを変えてお試し頂くなどの対応が必要かと思います。

コマンド操作の使用例

SDカード内のファイルを読む場合には、1つのコマンドを実行するだけでなく、いくつかのコマンドを実行して行う必要があります。
本項では操作方法の基本を説明します。

■既存のファイルを開き、データを読み込む場合

【操作の手順概要】
①SDカード内のファイルを空きハンドルに開く
②ファイルハンドルを指定してデータを読み込む

【実際のコマンド操作】
SDカードに "ABC.LOG" ファイルがある場合を例にします。
"ABC.LOG"ファイル内には下記のようなデータが書き込まれているとします。

※各行の文字列はCR(0x0D)で終端されています。
※MSC-MOD60からの戻り値は斜体で記載しています

```
O 1R>ABC.LOG{cr}
!00
R 1Z>65{cr}
!00
DATA            NAME            TIME
10/15           K.Kashima       10:00
10/15           T.Ueda          09:50
10/16           J.Inoo          10:30
10/16           H.Kawaka$00000065
!00
```

0x65(101バイト)バイトのデータがABC.LOGファイルから読み取られます。

■新規にファイルを作りデータを書き込む場合

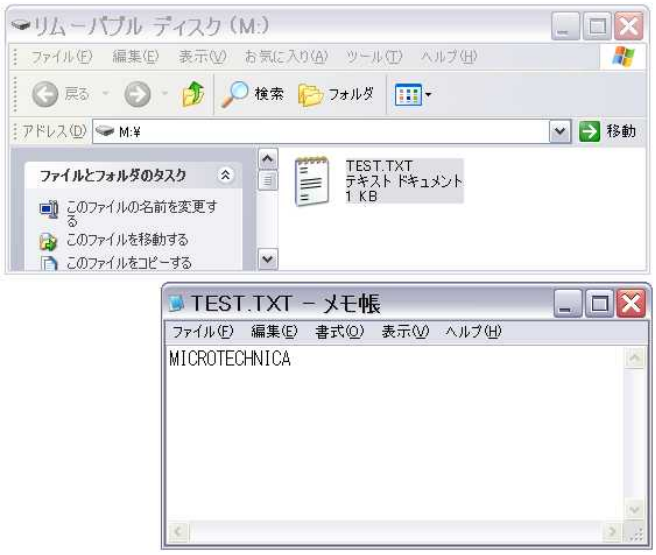
【操作の手順概要】
①ファイルハンドルにファイルを新規作成する
②作成したファイルにデータを書き込む

【実際のコマンド操作】
SDカードに "TEST.TXT" ファイルを作ります。そのファイルにデータとして"MICROTECHNICA"と書き込みます。

```
O 1W>TEXT.TXT{cr}
!00
W 1>D
!00
microtechnica{cr}
$0000000D
!00
!00                              ※1
C 1{cr}
!00
```

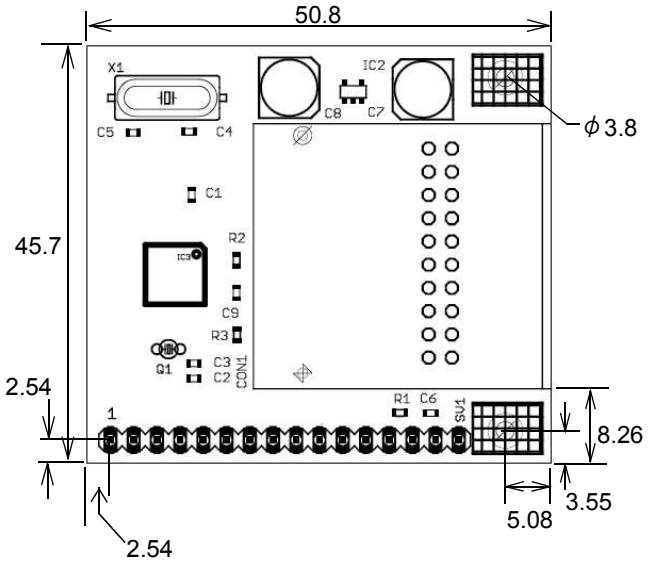
※1:このACKは、送信したデータ(文字列)の後ろがCRで終端されていたため、このCRに対応して返ったものです。データをCRで終端せずに指定したデータサイズで送ると、ACKが1つだけ返ります。

このUSBメモリーをパソコンで見ると下記のようになっています。



"TEST.TXT"が新規に作られ、ファイル内には、MICROTECHNICAとデータが書き込まれています。

外形寸法



単位はmmです。
インターフェイスピンは2.54mmピッチです。
基板厚は1.6mmです。
MMC又はSDカード挿入時は基板から約5.7mmカードが飛び出します。

※寸法は使用される部品によって予告無く変更されることがあります。

使用上の注意

MSC-MOD60を使用するに際して、必ず下記の注意事項をお守りください。

①MSC-MOD60を使用するに際し、当方は明示的及び潜在的な使用したことによる危険性や、不確実性については予見することができません。使用する際には、お客様の責任においてこの製品を正しくお使いいただけますようお願い致します。

②MSC-MOD60は、MMCやSDカード対してデータを記録したり、データを読み込むことのできるモジュールですが、データ書き込みや読み込みの信頼性は一般的な使用の範囲に限定されます。本製品を宇宙、航空、原子力、燃焼制御、運輸、交通、各種安全装置、ライフサポート関連の医療機器等のように、特別な品質・信頼性を要求され、その故障や誤動作が直接人命を脅かしたり、人体に危害を及ぼす恐れのある用途向けには使用できません。

③MSC-MOD60では様々な外的要因等によって、データを正しく書き込めなかったり、読み込んだデータに誤りがある場合などがあります。本製品を使用することによって生じた、もしくはこれに関連するいかなる直接・間接損害、懲罰的損害、その他データの破損や消失等を含むいかなる損害、損失についても、当方では一切責任を負いかねます。あらかじめご理解とご了承頂けますようお願い致します。

④本製品を使用した製品等を製造させる場合には、様々なフェイルセーフ機能(安全設計)を施して頂き、十分に機器のテストをした上で運用されますようお願い致します。また、データの損失や予期しない事態に備え、データのバックアップを行って頂けますようお願い致します。

製品の技術的なサポートについて

本製品の技術的なサポートは製品の開発元、米GHI Electronics社が直接行います。技術的なサポートが必要な場合には、開発元へ直接ご連絡頂きます。当方での技術的なサポートは致しておりません。

技術サポートはすべて英文となります。当方(日本)での技術サポートは行っておりません。あらかじめご了承頂けますようお願い申し上げます。なお、日本語マニュアル及びFAQにつきましては、当方のサイトより最新の情報をご提供致します。

本製品の開発元での型式は、"uALFAT-SD"です。本製品には、同社のワンチップFAT搭載CPUのuALFATが使われております。お問い合わせの際には、uALFATのお問い合わせとしてご連絡ください。
米GHI社では電話又はメールによる技術サポートを行っております。いずれかの方法にてお問い合わせ頂けますようお願い致します。
なお、サポートはいずれも技術者専門となっておりより技術的なご質問をするためのものとなっております。マニュアルに既に記載されている内容などについては回答が得られない場合がございます。

ghielec@ghielectronics.com

上記アドレスにご使用製品名とご使用者様の会社名、学校名、お名前、メールアドレス、貴社のWEBサイトのURL(学校の場合には学校のURL)をご記入頂き、メールをお送り下さい。追ってサポートスタッフから返信があります。文面には下記のようにご記入下さい。

『I would like to make contact with someone in charge of technical support. Please forward your reply to this email address.』

なおご質問事項はできるだけ使用環境、接続状況、エラーコードの内容など具体的にお書き下さい。
お電話の場合には、営業時間内におかけ下さい。

+1-586-693-2696 (米国の国番号は1です)

※通話には国際電話料金がかかります。また時差がありますのでご注意ください。(Officeは10:00AM～5:00PMまでです)

主な仕様

電源電圧:	DC3.3V(±5%以内)
消費電流:	フルパワーモード時 約38mA
	省電力モード時 約8mA
	ハイバネートモード時 約1.2mA
動作環境:	-10℃～70℃ (動作保証範囲)
対応SDカード規格:	SDカード
対応ファイルシステム:	FAT16 、FAT32
シリアル通信方式:	非同期式シリアル通信UART
	同期式シリアル通信I2C
信号電圧レベル:	LVTTL(但しTTL耐性あり)
CPU:	uALFAT (GHI Technology社製)
開発元:	米GHI Electronics社
開発元型式:	uALFAT-SD
生産国:	米国

microtechnica

マイクロテクニカ
〒158-0094 東京都世田谷区玉川1-3-10

(C)2009 Microtechnica All rights reserved